Natural Language and Multimodal Text-Tabular Explanations using Large Language Models and SHAP

JAMES BURTON

Natural Language and Multimodal Text-Tabular Explanations using Large Language Models and SHAP

James Burton

A thesis presented for the degree of
Doctor of Philosophy

Department of Computer Science
Durham University
United Kingdom

7th September 2024

Natural Language and Multimodal Text-Tabular Explanations using Large Language Models and SHAP

James Burton

Abstract

Explainability is a critical pillar of responsible AI, however, there remain several unexplored areas in the field. This thesis focuses on two key areas to enhance interpretability for the end user: utilising large language models (LLMs) to describe, in text, what the data in a table is showing, and creating and applying a new framework to applying SHapley Additive exPlanations (SHAP) explanations to text-tabular datasets. Firstly, this thesis introduces the novel task of taking a set of performance metrics - such as accuracy, precision and F1 score - and fine-tuning LLMs to explain this information. To do so, we collect and provide a dataset and experiment with a deep encoding of the metric information to enable clearer comprehension of the data table. The second chapter extends the text generation approach to explain classification decisions, using a second novel dataset of expert-written explanations to explain a numerical explanation: a set of feature importance values indicating which input the underlying model found most important to the decision. In fine-tuning LLMs on this dataset, experimenting with augmentation and a simplified question-answer task, we demonstrate the capacity to generate understandable and accurate natural language explanations. Further capitalising on the theme of explainability across multiple modalities, this thesis provides a solution to the inability to generate a numerical explanation for text-tabular datasets. Specifically, this thesis proposes a novel multi-modal masker that facilitates the production of SHAP values for any text-tabular dataset, for any method of combining the two modalities. In an extensive analysis, this thesis reveals the issues that arise when adapting the multi-modal dataset to a single modality (text) and applying the existing unimodal masker. Subsequently, we examine the impact that combination strategies and language models have on

SHAP values. Finally, we apply the proposed method to a veterinary dataset, using the generated explanations to carry out a deep-dive on which features models found most important and the reasons why PetBERT, an LLM pre-trained on a veterinary corpus, performs better than BERT, a general LLM.

Acknowledgements

A PhD is a true journey, one that, for me, began solo, locked in a covid-enforced isolation attempting to come to grips with a new field. It's been a wild ride full of (mostly) ups and (only a few) downs, but now coming to the end of and having space to reflect, I feel immensely grateful for the brilliant people that I have been surrounded with and who, without, this would not have been possible.

My principal thanks go to my supervisor, Noura Al Moubayed. Your guidance, warmth and ideas time and time again kept me going and guided me through to the sunlight uplands of PhD Year 3 and beyond. There are times I didn't believe you when you would say it would all work out fine, but here we are, so I wholeheartedly accept your smiling *I told you so*!

I extend a warm thank you to Amir Enshaei and Bashar Awwad Shiekh Hasan for your support, advice and supervision as well as all the other staff at Caspian for always making me feel welcome.

To Sean, whose jokes kept me coming back to the office and whose disapproving eyebrow raises kept me from leaving, I thank you. For making me feel part of the team and for countless hours of Zoom-based guidance, I thank Tom W. For more tech support than you ever signed up for I thank Tom H. Thank you Essel, for your collaboration and for graciously fielding my endless stream of questions and thank you to my other Computer Science colleagues, Matt, Dan, Patrick, Nour, Dean for your inspired ideas and lunchtime debates

To my friends in Durham, particularly Alan, Arnau, Carol, Chiara, Kev, Pietro and Sean, I thank you for making my time here one to be treasured, full of laughs and adventure, and one I'll never forget. I'm truly thankful for this time we have had together and the friendships we've made.

To Jen, thank you for being my biggest cheerleader, my partner in crime, and everything in between.

Finally I would like to thank my sisters and my parents for your unwavering support and consistent sound advice, despite me never quite finding to words to explain exactly what I do!

Contents

Declaration

The work in this thesis is based on research carried out at the Department of Computer Science., University of Durham, England. No part of this thesis has been submitted elsewhere for any other degree or qualification, and it is the sole work of the author unless referenced to the contrary in the text.

List of Figures

List of Tables

Chapter 1

Introduction

The advent of machine learning (ML) has catalysed an era of unprecedented computational innovation, transforming industries, revolutionising research methodologies, and altering the way we interact with technology. Machine learning algorithms now curate our digital experiences, inform critical decisions in healthcare, optimise logistics in transportation, and play an increasingly significant role within various other domains. These algorithms can now perform tasks ranging from the trivial, like recommending a movie, to the monumental, like assisting in diagnosing complex diseases.

In recent years, perhaps nowhere has the rapid development of ML been more evident than in the field of Natural Language Processing (NLP). NLP advancements have enabled functionalities such as instantaneous language translation, sophisticated chatbots, and advanced text analysis models that mimic human-like language understanding. As readily available tools such as ChatGPT demonstrate the ability to generate coherent and contextually relevant text, the applications of NLP continue to grow, penetrating both general-use technology and specialised sectors.

However, the remarkable leaps in machine learning capabilities come with newfound responsibilities. Among the most pressing is the notion of explainability: an acknowledgement that for ML to be responsibly integrated into society, its decisions and processes need to be transparent and interpretable. The foundation of this thesis rests upon the premise that explainability is not merely an academic fancy but a key tenet of responsible AI development and deployment. In areas where decisions have significant repercussions, such as healthcare, finance, and law, the ability to explain an ML model's reasoning is not just beneficial but often necessary for legal, ethical, and practical reasons.

Interpretability in machine learning is not only about understanding why a prediction was made but also how we can understand the results that are being presented to us. Machine learning is a complex field, often requiring background knowledge to comprehend the given results. This thesis aims to use the capabilities of language models to provide additional interpretability detail as well as to extend existing explanation techniques to new domains, all under the banner of bringing explainability to a wider audience in a manner that is accessible and understandable.

1.1 Motivation

A common trait for many a researcher, but one that is no less true for me, is an intellectual curiosity, a desire to figure out why something works and performs the way that it does. I have also found a certain joy in teaching, helping people come to grips with a new topic and helping them understand something complex. Simply put, I enjoy figuring things out and then communicating them in a way for others to understand. I believe this was a key part of my underlying motivation and why explainability has been a consistent theme in each bit of work I have done here at Durham.

Aside from explainability, another key development throughout my time spent at Durham has been the rapid transition to and development of deep learning and large language models (LLMs) in particular. I wanted to work in both of these fields and in this thesis, I find the niche of the two: firstly using language models to produce text that is itself an explanation of a different problem, and subsequently developing an explainability method to explain language models in the context of text-tabular modelling. The field of explainability is no longer new, therefore I chose to operate in the niche of multi-modal machine learning and explainability. Data-to-text tasks have models producing text that require comprehension of an input table, such as question-answer or summarisation tasks. I sought to operate in this space, utilising large language models to do so.

This thesis can be split into two separate, but connected parts: first I use language models to explain (in the literary sense) what is being shown by a numerical table, first narrating a table of performance metrics and second a set of feature importance values. The aim is to augment what is already there, giving users extra context in order for them to better understand what they are being shown. Secondly, I work with explainability in the machine learning context. Proposing a novel framework for text-tabular datasets, I make it possible to generate SHAP explanations for any model, for any combination method for the first time in this multi-modal context. In doing this I hope to move the dial on explainability one step further, allowing end users and machine learning practitioners alike to understand what's going on that little bit easier.

1.2 Research Questions

To summarise, the overarching goal is to bring explainability to more places, making it easier for people to understand machine learning outputs, but also to bring it to more places. Specifically, the two research questions that this thesis aims to answer are:

1. **Research Question 1**: This study aims to answer the question, to what extent can large language models be trained to produce text to enhance the understanding of a machine learning problem? Language models should be trained to produce text that complements the machine learning output and provides an additional manner in which the user can understand the problem

2. **Research Question 2**: Secondly, this study aims to answer the question, considering there is currently no way of producing SHAP explanations for text-tabular data, can a method be devised to bring SHAP explanations to this domain? The goal is to produce SHAP explanations for any text-tabular problem, no matter the dataset or experimental setup.

1.3 Publications

Each project contributing to this thesis has been submitted for publication or published in peer-reviewed journals or conference proceedings. These are detailed below:

- **Generating Textual Explanations for Machine Learning Models Performance: A Table-to-Text Task**
  - James Burton, Isaac Ampomah, Amir Enshaei and Noura Al Moubayed
  - Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC 2022)
  - Chapter 3
- **Natural Language Explanations for Machine Learning Classification Decisions**
  - James Burton, Amir Enshaei and Noura Al Moubayed
  - 2023 International Joint Conference on Neural Networks (IJCNN)
  - Chapter 4
- **SHAP Explanations for Multimodal Text-Tabular Models**
  - James Burton and Noura Al Moubayed
  - Under Review at Nature Scientific Reports
  - Chapter 5

- **Explainable Text-Tabular Models for Predicting Mortality Risk in Companion Animals**
  - James Burton, Sean Farrell, Peter-John Mäntylä Noble and Noura Al Moubayed
  - Nature Scientific Reports
  - Chapter 6

1.4 Thesis Structure

This thesis begins with **Chapter 1**, this chapter, where explainability and large language models are introduced, setting the scene for the rest of the thesis. In this chapter, a motivations section is laid out to provide an indication as to why these research questions have been chosen and details of publications are listed.

Chapter 2 is an extensive review of the relevant literature. It introduces Natural Language Processing and discusses the language models and evaluation metrics used in this thesis. Subsequently, it will cover text-tabular machine learning and explainability, giving a background on specific techniques used with a deep dive on SHAP, a key tenet of Chapters 5 and 6.

In Chapter 3, I propose a new task: take a set of metrics detailing a machine learning model's performance (including, for example, accuracy, precision and F1 score) and train a language model to describe the model's performance in text, with the aim of making it simpler for non-experts to understand what the metrics are describing. I collect a new dataset -collected from computer science experts - and fine-tune pre-trained language models to generate informative and analytical passages of text. I propose an improvement to the model's structure, the *Metric Processing Unit*, to more deeply encode the metric information and provide higher quality textual explanations.

Chapter 4 introduces another new task, once more with the goal of translating technical information into an accompanying passage of text. In this case, the task begins with feature importance values (the output of a certain category of explainability techniques) that numerically describe how important each input feature was to the outcome of a machine learning classification model. A second novel dataset is collected to train language models to translate the numerical explanation - the feature importance values - into a textual explanation. Finally, the numerical explanations are probed more directly by creating a question-answer task with a large, synthetically generated training set.

Chapter 5 continues with the theme of explainability, but more directly. The research gap in explainability for text-tabular models is addressed, bringing to light the errors that occur when coercing the multimodal input to fit the existing unimodal text masker and a novel multi-modal masking framework that extends SHAP to text-tabular datasets is proposed. With this framework, it is possible to generate SHAP feature importance values for any dataset, any text model and any method of combining the two modalities. Extensive experimentation is conducted across all three of these domains as well as an investigation into whether unimodal errors - focusing on the attributing of importance to static text template values - are more prevalent for certain modalities or other feature properties.

In Chapter 6 the multimodal masking framework is applied to identify risk factors of pet mortality in a corpus of veterinary electronic health records. Once more, different combination methods and text models are tested, in particular comparing BERT with PetBERT, a model specifically pre-trained for veterinary data. An analysis of the similarities and differences is presented, then opening it up to a more granular level, this chapter looks at the individual words, phrases and tabular feature values that are most influential in mortality prediction.

Chapter 7 brings the thesis to a conclusion by looking back at the work as a whole. It puts forward the case for continued explainability research and scrutinises each part of the thesis to make recommendations for future strategies for data-to-text applications.

Finally, Chapter 8 is an appendix containing graphs and figures from Chapter 5 that are not vital for the story of the Chapter 5, but are included to be complete.

1.5 Contributions

A machine learning model's performance is usually assessed by a set of metrics. Some metrics are intuitive, but others are less easy to understand, especially for an end user. With this in mind, in Chapter 3 a new task is created aiming to bridge this gap and make it easier for people to understand a table of machine learning metrics by training language models to produce a textual summary to complement the set of metrics. Other data-to-text tasks exist, but this is a novel task. A large part of the contribution for this chapter is the dataset of table-text pairs, however, it also includes experiments with augmentation and the introduction of a way to more deeply encode the metric information.

Chapter 4 continues to address the first research question of making machine learning outputs more understandable by training large language models. The same comparisons as the last chapter apply here also: other data-to-text tasks exist but this is a novel task. As far as we are aware, no other work has been done to take the feature importance scores of an explainability output and translate it into something more understandable and approachable. Alongside the novelty of the task, another novel dataset of numerical-textual explanation pairs forms a large part of this chapter's contribution.

Chapter 5 provides arguably the most significant contribution of this thesis. As Chapter 2 will discuss in more detail, in recent years there has been research done to bring explainability to the text-image domain, but the text-tabular domain has been left untouched. This chapter ports SHAP to this domain, allowing SHAP explanations to be produced for text-tabular datasets for the first time. This is a significant contribution as now it opens the door to a vast range of possible analyses in an area where it was previously not possible to get a joint understanding as to why a prediction had been made.

Chapter 6 provides a deep dive into a veterinary text-tabular dataset. As Chapter 2 will discuss, previous work had looked at predicting pet mortality given a set of tabular features (such as age and breed) and textual features (such as the free-text clinical notes). It was possible to get an answer to determine which text or tabular feature was the most important when taken in isolation, but never in combination. With the tool introduced in Chapter 5, we can now see where each text feature ranks alongside each text feature as the most important, as it should be considering our predictive models are using both feature types together.

Chapter 2

Literature Review

This chapter will give a background on the relevant information for this thesis. It will introduce Natural Language Processing and LLMs, including specifics on the models used throughout this study, as well as the evaluation metrics that are used in later chapters to assess said models. Text-tabular machine learning is also reviewed, covering techniques, terminology and specific tasks. It finishes with a review of explainability, aiming to give a background on the subject and cover the explainability methods that are used in later sections, with a special focus on SHAP.

2.1 Natural Language Processing

Natural Language Processing (NLP) is a distinct and significant branch of artificial intelligence that covers the communication, understanding and processing of Natural Language by computers. Emerging as a field in the 1950s (Johri et al., 2021), first aiming to produce a machine capable of translating foreign language automatically, NLP has expanded into many aspects of day-to-day life, from grammar checking to AI virtual assistants such as Siri and Alexa. Beyond consumer products, NLP also covers domains such as sentiment analysis (Dang et al., 2020), document summarisation (Gupta and Gupta, 2019) and text classification (Minaee et al., 2021), indeed any task involving text, whether that be written or spoken.

The difficulty of NLP tasks comes from the fact that computers require information to be structured, whereas language is just the opposite. Language has an inherently complex nature: it can be ambiguous; words can carry multiple meanings; there can be wordplay, idioms, sarcasm and humour, all of which make understanding the true meaning of text difficult for a machine only built to work with 0s and 1s.

2.1.1 Modelling NLP

Before 2017, NLP tasks were usually tackled using Recurrent Neural Networks (RNNs) (Rumelhart et al., 1986), which, by design, process sequences of data by feeding the outputs of a neuron back in as an input in a recurrent fashion. RNNs were known to suffer from the *short-term memory problem* so varieties of RNN called Long Short Term Memory networks (LSTMs) (Hochreiter and Schmidhuber, 1997) and later Gated Recurrent Units (GRUs) (Cho et al., 2014) were developed to tackle this problem and look to give the RNN a longer-term memory.

LSTMs learn to remember what is important and forget what is not as they pass over the input sequence, updating its internal hidden state, which is then passed in alongside the next input (the next word) in the sequence. Once the RNN has reached the end of the sentence, the hidden state then contains information about the whole sentence and can then be used as input for a sequence-to-sequence task, such as translation. Sequence-to-sequence tasks take the input of the initial RNN (the encoder) and pass it into another RNN (the decoder). The decoder then uses the encoded vector and outputs the target sequence one token at a time.

In 2017, the seminal ‘Attention is All You Need’ (Vaswani et al., 2017) was published, introducing the transformer, which profoundly changed the NLP field. Instead of being limited to one encoded vector containing all of the information about the encoded sentence, the transformer utilised multi-head self-attention to learn how each input in a sequence relates to each other, which is then combined with feed-forward layers to produce a highly contextualised encoding of the input sequence. Transformers do not suffer from the same short-term memory problem as RNNs as they process all of the inputs at the same time instead of sequentially and are, therefore, able to parallelise and stack encoders one on top of each other to learn deep relations between the input words.

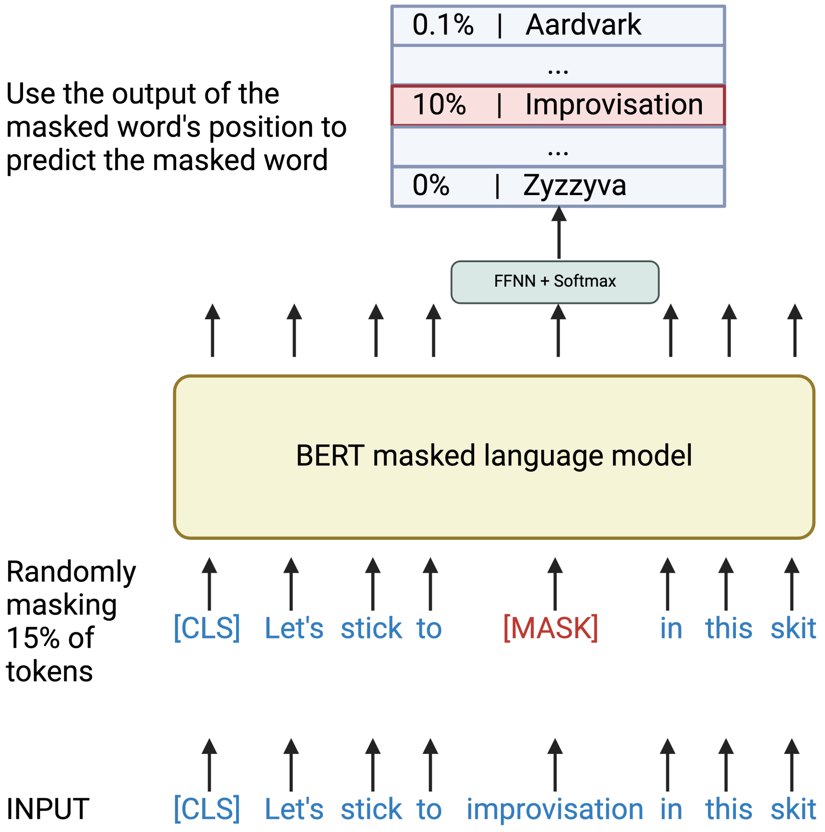
2.1.2 Deep Learning Models in This Study

In this thesis, a selection of different language models are used. This subsection will aim to give a background of those models and briefly describe how they work. This thesis involves direct experimentation with all language models discussed here.

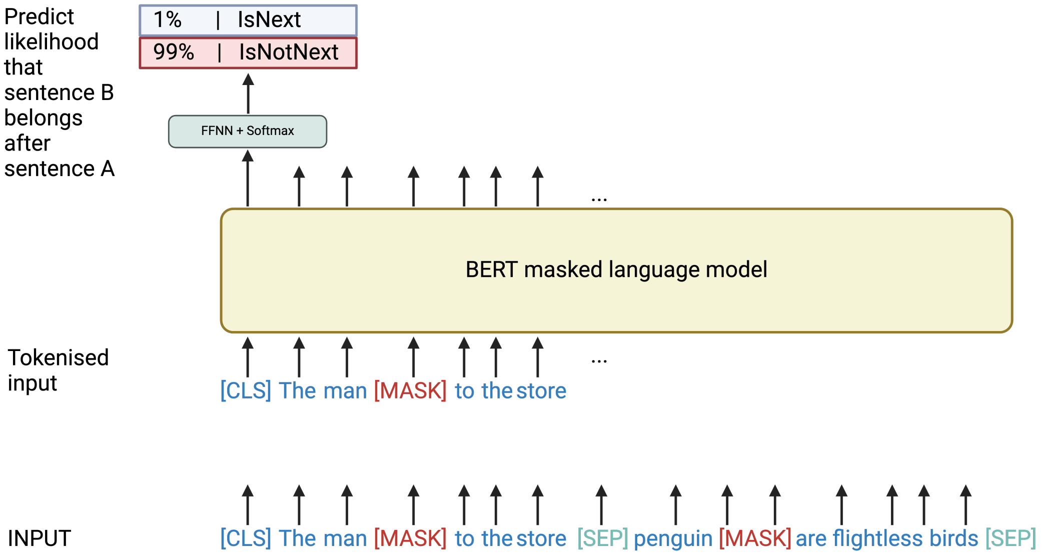
2.1.2.1 BERT

Bidirectional Encoder Representations from Transformers (BERT), as introduced by Devlin et al. (2018), represents a significant advancement in language modelling. BERT is constructed by stacking a series of transformers, which are instrumental in its ability to process language data efficiently. Another key advantage of BERT is in its use of self-supervised learning; whereas an LSTM needs to be trained from scratch each time, BERT is pre-trained with a Masked Language Modelling (MLM) and Next Sentence Prediction (NSP) objective to allow it to gain an understanding of linguistic rules and context.

A major benefit of BERT is the practicality it offers for adaptation. Since the model is pre-trained to understand the general structure and flow of language, typically, one only needs to train the topmost layers of the network to adapt it to a specific task. This secondary training phase is known as fine-tuning the model. Fine-tuning is significantly faster and more resource-efficient than training a new language model from scratch.


0.1% | Aardvark
...
10% | Improvisation
...
0% | Zyzzyva
Use the output of the masked word's position to predict the masked word
FFNN + Softmax
BERT masked language model
Randomly masking 15% of tokens
[CLS] Let's stick to [MASK] in this skit
INPUT
[CLS] Let's stick to improvisation in this skit

(a)


Predict likelihood that sentence B belongs after sentence A
1% | IsNext
99% | IsNotNext
FFNN + Softmax
...
BERT masked language model
Tokenised input
[CLS] The man [MASK] to the store
...
INPUT
[CLS] The man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]

(b)

Figure 2.1: BERT's MLM (a) and NSP (b) objectives

2.1.2.2 DistilBERT

Powerful models are, of course, useful, but another important consideration is how resource-intensive they are. Sanh et al. (2019) proposes DistilBERT, a model that is 40% smaller and 60% faster than BERT, but still retains 97% of the language capabilities. DistilBERT is trained to distil the knowledge contained by BERT into a smaller structure. It does this by using a triple loss, combining the same MSM objective with two more objectives: a cosine loss to match BERT's hidden representation and a distillation loss to mirror BERT's predicted probabilities. DistilBERT does not use BERT's NSP objective.

2.1.2.3 DistilRoBERTa

A Robustly Optimised BERT Pretraining Approach (RoBERTa) (Liu et al., 2019) is an iteration of BERT which modifies key parts of the training procedure, removing the Next Sentence Prediction objective, training with larger mini-batches over more data and dynamically altering the masking. In exactly the same fashion as with DistilBERT, DistilRoBERTa (Sanh et al., 2019) was created as a distilled, streamlined version of the full model, utilising a smaller footprint to emulate the performance of RoBERTa.

2.1.2.4 DeBERTa

He et al. (2021b) propose a novel architecture, innovating on BERT and RoBERTa by incorporating two key modifications. Firstly, the previously unified content and position representation are separated into two, thereby serving the model with a clearer understanding of the input text. Secondly, the MLM objective is adjusted so as to take into account the absolute position of the masked word in the sentence. The resulting model is named Decoding-Enhanced BERT With Disentangled Attention, or DeBERTa.

The authors subsequently released an update to DeBERTa (v3 (He et al., 2021a)), where they utilised two further improvements. ELECTRA Clark et al. (2020) - a model not used in this thesis - swapped the MLM objective for Replaced Token Detection (RTD), where instead of estimating a masked word, the model is now concurrently trained as a discriminator and must identify whether a token is original or has been replaced. In this update to DeBERTA, the authors make the swap from MLM to RTD but alter the training procedure slightly: the generator, which is in charge of initiating the token corruptions, and the discriminator are now trained with separate embeddings to avoid their distinct training objectives clashing. This thesis uses this latest update.

2.1.2.5 T5 model

T5 (Raffel et al., 2020) is a transformer-based model trained in a multitask fashion on a variety of unsupervised and supervised NLP tasks, including summarisation, classification, and translation. This neural model treats all text-based language problems as a text-to-text generation task (Raffel et al., 2020). The training process is shown in Fig. 2.2.

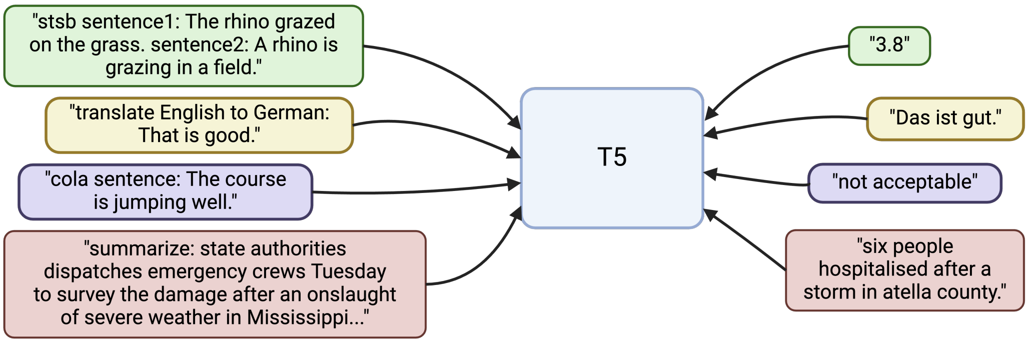

"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"translate English to German: That is good."
"cola sentence: The course is jumping well."
"summarize: state authorities dispatches emergency crews Tuesday to survey the damage after an onslaught of severe weather in Mississippi..."
T5
"3.8"
"Das ist gut."
"not acceptable"
"six people hospitalised after a storm in atella county."

Figure 2.2: Diagram of the T5 training process, all tasks are converted into text such that the same model can be used.

2.1.2.6 BART model

Bidirectional and Auto-Regressive Transformers, or BART (Lewis et al., 2020), is a transformer-based denoising autoencoder. During pre-training, BART is fed input text that has been corrupted in various ways, expanding on the token masking used in BERT. In addition to token masking, BART also applies sentence shuffling, token deletion, document rotation and span infilling, providing a more varied and challenging objective. The model must reconstruct the original, uncorrupted text. The wider variety of tasks forces BART not only to learn the relationship between neighbouring words, but also how sentences within a document relate to each other. For this reason, BART can be highly effective in summarisation and text-generation tasks.

2.2 PetBERT and SAVSNET

The previous section described some general-purpose language models that are used throughout this thesis. In Chapter 5, SHAP for text-tabular datasets will be presented. In Chapter 6, SHAP will be applied to a specific text-tabular dataset to explain the predictions of a trained model on a particular task. This section will begin by detailing the particular veterinary dataset (SAVSNET) that is used. It will then cover the trained language model (PetBERT) that forms the base of all the experiments. Finally, this section will give a thorough description of the task that is being explained, namely using the information in SAVSNET to predict the one-month mortality of a given companion animal.

2.2.1 SAVSNET Dataset

Electronic health records have been collected since March 2014 by SAVSNET (Sánchez-Vizcaíno et al., 2015), comprising a sentinel network of 253 volunteer veterinary practices found across the United Kingdom. Generally, veterinary practices with practice management software compatible with the SAVSNET data exchange are recruited based on convenience. Within these participating practices, data is collected from each booked consultation (where an appointment has been made to see a veterinary practitioner or nurse). All owners attending these practices can opt out of data collection at the time of consultation. Data is collected on a consultation-by-consultation basis and includes information such as species, breed, sex, neuter status, age, owner's postcode, insurance and microchipping status and, crucially to this study, a free-text clinical narrative outlining the events that occurred within that consultation. Appended to all the SAVSNET Electronic Health Record (EHR) datasets are high-level International Classification Disease 11 (ICD) codings. These syndromic labels can provide a broad overview of the themes within the clinical narrative, a free-text field.

300 manually labelled records were used to train binary classifiers, one classifier for each of the 20 possible labels. These binary classifiers were subsequently applied to the remainder of the dataset (Farrell et al., 2023a). Sensitive information, such as personal identifiers, was cleaned from the data. SAVSNET has ethical approval from the University of Liverpool Research Ethics Committee (RETH000964).

2.2.2 PetBERT

PetBERT (Farrell et al., 2023a) is based on the BERT-base model, which - as explained above - was previously pre-trained to perform two tasks simultaneously: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). PetBERT underwent a similar training process: starting from the pre-trained BERT-base model, PetBERT was additionally pre-trained on a large dataset of over 500 million tokens from the SAVSNET first opinion veterinary corpus, exposing it to clinical language used in veterinary contexts. Note that it is still referred to as pre-training - as opposed to fine-tuning - because the objectives are still MLM and NSP, not a downstream task. Furthermore, all parameters are trained, unlike fine-tuning, where typically only the output layer is trained.

2.2.3 Mortality Prediction with PetBERT on SAVSNET

In Farrell et al. (2023b) the authors fine-tuned PetBERT in order to predict whether or not a companion animal would die, based on a particular consultancy or set of consultancies in the SAVSNET dataset. To do so, labels needed to be created. To curate the datasets for training the initial component of the predictive mortality models, the authors searched for narratives containing references to death or euthanasia. This search used a generalised Python regular expression to identify pertinent terms, including ‘euthanasia’, ‘put to sleep (PTS)’, and ‘died’. The detailed regex pattern is provided below in Equation 2.1.

$$\textit{euth/dead/died/pts/put to sleep/pento/doa/crem/burial/bury/qol/quality/ashes/scatter/casket} \tag{2.1}$$

Subsequently, random sampling was performed to select 250 cases that were suspected to involve mentions of death or euthanasia. These selected cases underwent manual inspection to validate whether they conformed to the predefined case definition of ‘declaration of death occurring within the consultation’. Notable instances of false positives included conversations of potential future euthanasia events or instances where euthanasia was discussed in an advisory context by the attending practitioner. Instances where the euthanasia event did not occur within the same consultation were excluded or used as the controls in equal proportion to the number of cases.

A semi-supervised teacher-student model approach was adopted in line with the methodology employed by Zeki et al. (2019). This approach used a small subset of manually annotated records to train a small binary sequence classification model, which achieved an F1 Score of 98.3% on the test set. This model was subsequently applied to the entire dataset to identify animals meeting the criteria. To ascertain the effectiveness of this extraction method, a random sample of 200 records was independently reviewed by a practising clinician to validate the model’s performance and suitability for the continuation of the study.

For the animals identified to have died by the above binary sequence classification task, the authors took the consultation preceding the declaration of death. To create a balanced dataset, animals stated to be alive by the binary sequence classification model were pulled at equal quantities to the number of animals that had died. Narratives for cases where the animal had only a single narrative in its history (the one detailing its death) were discarded. Within both the case and controls, where incomplete data exists, such as missing breed, age, sex, geographical information, or where an animal appeared in both case and control datasets, these records were also deleted. All high-level ICD codings that the animal has previously amassed were summed together. The frequencies of each ICD coding were used to represent each animal's approximate clinical history and maximise the availability of tokens for the penultimate clinical narrative for PetBERT.

2.3 Evaluation Metrics

This thesis uses several automatic metrics to judge the quality of generated text. The metrics employed to assess the quality are BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), BLEURT (Sellam et al., 2020) and PARENT (Dhingra et al., 2019). This section details how each metric is calculated. BLEU, and METEOR compute the surface-level similarity between the generated texts and only the human (reference) texts.

2.3.1 BLEU

The BiLingual Evaluation Understudy, known as BLEU, is a relatively old method that is still commonly used to this day. It is designed to be used for translation, assessing how close a translated text is to the original, high-quality reference text. The idea is to calculate the n-gram overlap, that is, the number of times n-tokens in the candidate match that in the reference. In order to penalise candidate sentences that are too short, a brevity penalty, *BP*, is added.

Mathematically we have

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.2}$$

$$BP = \begin{cases} 1 & \text{if } c > r \\ \exp\left(1 - \frac{r}{c}\right) & \text{if } c \leq r \end{cases} \tag{2.3}$$

N is the number of n-grams, w_n is the weights for each n-gram precision. p_n is the modified n-gram precision, which is the number of correct n-grams in the candidate translation divided by the total number of n-grams in the candidate translation. c is the length of the candidate sentence and r is the length of the target sentence.

The values of the BLEU score range between 0 and 1 where 1 means a perfect match with the reference translation and 0 means no overlap. Note that the relationship between BLEU and human judgment is not linear, so small differences in BLEU score may not correlate to differences in translation quality.

The default n-gram for BLEU is 4 as this is the score with the highest correlation with human judgement (Papineni et al., 2002), and typically, a balanced set of weights is used such that $w\_n = 1/N$. This is used throughout this thesis. Therefore, the number of 1-gram overlaps, 2-gram, 3-gram and 4-gram overlaps are calculated, each weighted evenly.

As an effective metric for judging machine translation, a criticism of BLEU is that it is not well designed for other tasks. In a review of 34 papers over the last 15 years, Reiter (2018) conclude that decisions should not be based solely on BLEU as correlations between BLEU scores and human evaluations can vary. BLEU is optimising a ‘surrogate’ target, that being word overlap, which does not always align with the intended target.

2.3.2 BLEURT

The aim of BLEURT (Sellam et al., 2020) was to improve BLEU by drawing on the abilities of LLMs to better capture the finer differences between sentences; LLMs predict the scores of automatic metrics before being fine-tuned on human annotations to better mirror human preference.

BLEURT starts with a large corpus of sentence translations and then applies a selection of automatic metrics, including BLEU, ROUGE (Lin, 2004), BERTScore (Zhang et al., 2020), and back-translation. A BERT model, which has already been pre-trained with a language modelling objective, is further pre-trained to predict an amalgamation of the automatic metric set. Finally, this model is fine-tuned on a smaller dataset of human-annotated translations in order to align the metric better with human preference. In a sense, it is similar to the distilled models discussed in Section 2.1.2, where instead of predicting the output of other models, BLEURT is learning to emulate other metrics. One could also draw parallels to ensemble models (discussed later in Section 2.4.1) as a selection of outcomes are pooled into a single score.

2.3.3 METEOR

METEOR, an acronym for Metric for Evaluation of Translation with Explicit ORdering, was proposed by Banerjee and Lavie (2005). The goal was to address some of the issues that occur with BLEU, an inherently precision-based metric. First, the candidate and reference sentences are aligned, such that each unigram in the candidate sentences is matched to 0 or 1 unigram in the reference sentence. Whereas BLEU only counts exact matches, METEOR is slightly more sophisticated and will look for matches in three ways, moving on to the next only if it doesn't find a match in the previous. These modules are an exact matcher, a stemmer and a synonym finder. An example is shown in 2.1.

Table 2.1: Examples of pairs of words and how they will be mapped by each module of the METEOR matching process

Module	Candidate	Reference	Match
Exact	Walk	Walk	Yes
Stemmer	Walking	Walk	Yes
Synonymy	stroll	Walk	Yes

Unigram precision, P, is calculated as

$$P = \frac{m}{w_t} \tag{2.4}$$

Unigram recall, R, is calculated as

$$R = \frac{m}{w_r} \tag{2.5}$$

where m is the number of unigrams in the candidate translation that are also found in the reference translation, w_t is the number of unigrams in the candidate translation and w_r is the number of unigrams in the reference translation. Whereas BLEU only directly controls recall, METEOR uses a balance of both recall and precision:

$$F_{mean} = \frac{10PR}{R + 9P} \tag{2.6}$$

10 and 9 are optimised parameter values that were selected by Banerjee and Lavie (2005). METEOR controls fluency by also creating a penalty term, p, which is calculated by looking at how many matched 'chunks' that the candidate and reference sentences can be split into. As a formula:

$$p = 0.5\left(\frac{c}{u_m}\right)^3 \tag{2.7}$$

where c is the number of chunks and u_m is the number of mapped unigrams. Together we have the meteor score M as

$$M = F_{mean}(1 - p) \tag{2.8}$$

2.3.4 PARENT

Precision And Recall of Entailed Ngrams from the Table or PARENT (Dhingra et al., 2019) is a metric specifically designed for a data-to-text task. Dhingra et al. (2019) argued that BLEU and METEOR sometimes penalise generated text for including additional information, correct according to the table but missing in the reference text. In a translation task, it is right to have a gold standard reference text, but for a data-to-text task (the authors focus on captioning a table from Wikipedia), the provided text may not be the only correct way of describing the data. In response, they proposed PARENT, a data-to-text evaluation metric that takes into consideration the information present in the table.

Specifically, all information in the table is represented as tuples of entities, attributes and values. The final score is a combination of both precision and recall of the n-gram overlap between both the candidate and the reference texts and the candidate text and the table.

2.4 Text-Tabular Machine Learning

Data can take on many forms or ‘modalities’, from text to tabular, from audio to image. However, these distinct data types often require very different models to process their specific characteristics optimally. For instance, as discussed in Section 2.1.1, language is most often processed using transformer-based models. Tabular data, on the other hand, presents a distinctly separate challenge due to its structured nature, penchant for missing values and the need to manage both numerical and categorical data fields.

Tasks are often restricted to only tabular input or only text input. However, there are many instances where a dataset might be multi-modal; in other words, it may have inputs of both types of data. In practice, this would be a structured tabular dataset with unstructured text fields. A useful taxonomy outlining the stages of multi-modal modelling - and one that provides the language used later in this section - can be found in Sleeman et al. (2021). Briefly, it describes:

1. **Preprocessing and Feature Extraction**: Data is cleaned, and high-level features are extracted. This covers data augmentation, handling missing values and any other steps necessary to undertake pre-modelling.

2. **Data fusion**: Early, late or hybrid fusion describes *when* the combination of the modalities happens, but one also needs to outline *how* this is done. A simple but effective method is *concatenation*, where the representation of one modality is appended to the representation of another.

3. **Primary Learner**: The majority of the learning process happens here. This could happen jointly or independently, with a model trained separately on each modality.

4. **Final Classifier**: When modalities are trained separately, a final stage is needed to combine the results together. In a jointly trained model, the primary learner and the final classifier are one and the same.

2.4.1 Ensemble Learning

One common technique is to have the final classifier as an ensemble. Ensemble learning is a robust strategy for the combination of multiple models, either from the same or different modalities. It is based on a very human concept of the ability of a collection of individuals to outperform the singular, known as *wisdom of the crowd*, which can be traced back to Aristotle (*wisdom of the multitude*) and discussed more recently in Simoiu et al. (2019) and Yi et al. (2012). An often cited example by Galton (1907), who at a village fair collected the guesses of 787 people for the weight of an ox and famously observed that the median guess of 1,197 lbs was just a single pound away from the true value.

The driving principle behind ensemble methods is that while single individuals, or models, may have unique strengths and weaknesses, their aggregation can both amplify their predictive capabilities and mitigate the impact of their individual errors. Hence, an ensemble will perform best when the underlying models or *base learners* are diverse. Of course, models using different data sources (or modalities) fit this description.

Predictions from individual learners can be combined in various ways, such as a simple weighting, which is most suitable when model performance is comparable (Sagi and Rokach, 2018). A non-even weighting is suitable when performance differs, with one strategy being to assign weight based on the individual model's performance on a validation set (Opitz and Shavlik, 1995).

XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017) - two popular and effective methods - are based on an ensemble of simple decision trees, plus gradient boosting, which iteratively refines the predictions by focusing on correcting the errors made by previous trees in the sequence. These have shown to be effective on structured, tabular data, even outperforming deep learning models (Shwartz-Ziv and Armon, 2022) (Grinsztajn et al., 2022) or being more robust to less informative features (Hollmann et al., 2022).

2.4.2 Text Generation from Structured Data

For text-tabular datasets, a common natural language generation (NLG) task is to use the information in the table to produce an answer in a question-answer task (Yin et al., 2020; Bao et al., 2018), or generating summaries or captions (Suadaa et al., 2021a; Puduppully and Lapata, 2021).

Earlier work on data-to-text generation predominantly used rule-based methods (Goldberg et al., 1994; Reiter and Dale, 1997; Strauss and Kipp, 2008). These methods generate natural language text by employing linguistic rules and heuristics to select and populate pre-defined templates. However, a typical NLG system requires different sets of rules to perform content determination, text planning, sentence planning and surface realisation modules (Goldberg et al., 1994; van der Lee et al., 2017). This makes traditional NLG models difficult to maintain and less generalised.

Recently, leveraging deep neural methods for NLG has been shown to outperform existing rule-based methods (Wen et al., 2015; Liu et al., 2018; Puduppully et al., 2019; Parikh et al., 2020; Suadaa et al., 2021b). These models are usually trained end-to-end without the need for pre-defined linguistic rules. In broader terms, transfer learning has been shown to produce close to state-of-the-art performance for downstream NLP tasks with a limited amount of the dataset by utilising large pre-trained language models (Devlin et al., 2019; Radford et al., 2019). Furthermore, Peng et al. (2020); Chen et al. (2020b) argue that pre-trained language models (GPT-2 (Radford et al., 2019) and T5 (Raffel et al., 2020)) can indeed improve performance on structured data-to-text task.

2.4.3 Traditional ML Models as Primary Learner: Extracting Features from Text

For when using a model specifically suited for tabular data is desired, it is rare to use a separate transformer model for the text. Typically, methods will simply look to extract features from the text that are fed directly into a tabular model directly. For example, Shin et al. (2019) aimed to incorporate various sets of clinical notes, alongside structured tabular data. They compared a logistic regression model which took in all the tabular data as well as all of the text data (in a Term Frequency-Inverse Document Frequency, ie TF-IDF, form) as a baseline and found that using an ensemble, where logistic regression models were used to make predictions for each individual clinical notes/structured dataset, not only improved on the baseline, but also was able to handle missing information and capture information from long documents.

Khaleghi et al. (2021) predicts the surgical classification using unstructured surgical notes and procedure descriptions alongside structured tabular data such as patient age and case type to predict the set of procedures in the surgery room. As in Shin et al. (2019), the authors use TF-IDF, in this case, to extract the frequency of medical terms. In a similar task, Alekhya and Sasikumar (2022) used a fuzzy logic ruleset to extract features out of the text data, feeding it directly into an SVM alongside extracted tabular data for a healthcare dataset.

Azri et al. (2023) uses user information and tweets (containing text and images) to solve a rumour classification task. Once more, the text is distilled down to its component features, with the extracted information including length of tweet and number of exclamation marks. The authors experiment with late and early fusion to combine structured data, extracted text features (and image features extracted in a similar manner) and find the best performance with a stacking method from Wolpert (1992) (where a meta-model is used to predict from the predictions of base models).

2.4.4 LLMs as Primary Learning: Using Text Directly

In Gu and Budhkar (2021), the authors present a toolkit for using transformers with text and tabular data, illustrated in Figure 2.3. It is designed to be sufficiently general enough for implementing a number of different fusion methods and final classifiers including attention, concatenation of input representations and ensembling. The research is primarily to publish the framework, however, the authors do experiment with one regression dataset and two classification datasets.

Shi et al. (2021) conducts a more thorough evaluation, with the goal being to test Automatic Machine Learning (AutoML) strategies across a number of datasets, all containing text and tabular data. One simple early-fusion method the authors use is to convert all inputs to strings and train a text model. In this case, there is no need for separate models for each data source, and a single model is tasked

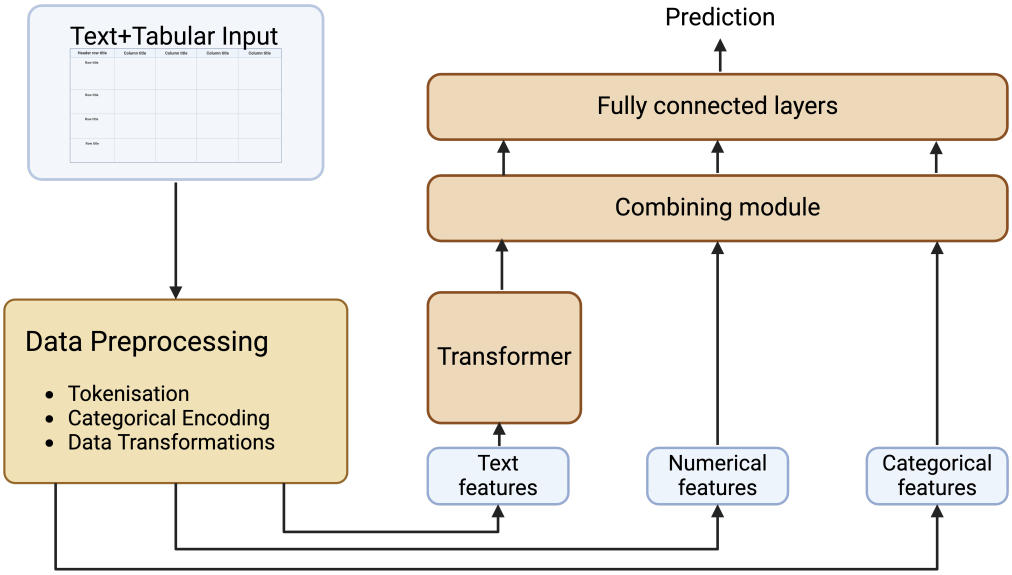


Figure 2.3: Text and tabular Toolkit

with learning the relationships between the inputs. In essence, this is what the T5 model, explained in Section 2.1.1, is designed to do. The authors test this all-text method alongside other methods, such as using the text embedding directly as a feature or a stack-ensemble where a meta-model is used to learn from tabular features and base-model predictions. In this work, the stack model performs best overall. This work is one of the few examples that use transformers (in conjunction with other models) to process the text information as opposed to extracting out tractable features to be used by ML methods.

2.5 Explainability

2.5.1 What is Explainability?

As noted by Vilone and Longo (2021), there is no consensus on how explainability should be defined; in this work, we use the concept of explainability to refer to how understandable the prediction of a given model is. It is the quality of how well the why question gets answered, i.e., why this model made this prediction. Which part of the input did it rely upon the most? Is there a bias in the model that we can reveal by asking the *why* question? Explainability is important, and it is a broad field. Firstly, a model can either be intrinsically explainable - a white box model - or not: a black box model. A linear model is a classic example of a white box model; we can look at the coefficients and the values to see exactly how the model arrived at the prediction. A simple decision tree is another example of a white box model. A neural network is the archetypal example of a black box model; we can check the calculations are correct, and we know in a general sense that the model used backpropagation to train, but we can gain no insight as to why specifically the model predicted the way it did. In the age of deep learning, most models are not inherently explainable; even decision trees lose their explainable quality once they become too deep or we add several decision trees in an ensemble to form a random forest.

For black box models, we require an additional step, applying an additional algorithm to elicit an explanation. These algorithms can be grouped; further, some can be used on only certain model types, others are agnostic to the model choice, some provide *global* explanations which describe the behaviour of the model on the whole, whereas others provide *local* explanations which explain the reason a particular instance was predicted the way it was. Danilevsky et al. (2020) focuses on the perspective of an end-user seeking to understand how the model arrives at its result. In their review, 46/50 of papers are local explanations and 4/50 are global.

Taking a step back from the intricacies of algorithm development, Miller (2018) took a different approach; the authors provide a comprehensive review of psychological literature so as to shine a light on what humans look for in an explanation from other humans and, therefore, erects a target for machine explanations to aim for. They highlighted that explanations should be *contrastive*: they are a response to a particular case asking what caused event A to happen *instead* of event B; this type of explanation is described as a counterfactual. Humans also prefer *selective* explanations, those that focus on a reduced number of causes; assembling a mental image of the problem becomes easier the fewer categories one has to keep account of. I suggest that this could be counter-productive. Humans prefer problems that are easier to solve (Murawski and Bossaerts, 2016). We are efficient creatures, and therefore, if we believe even complex issues can be explained away by an easy answer, we reduce our cognitive strain. However, this does not mean it is the right thing to do.

2.5.2 Why Do We Want Explainability?

As models have become more powerful and have been used for more and more things, concerns have been raised about the ability to justify the predictions that have been made. This is particularly the case when the stakes are high, such as in healthcare or legal scenarios where decisions can have serious implications. For example, racial bias was found in the COMPAS dataset (Angwin et al., 2022), a dataset that is used to predict the rate at which criminals re-offend. A model used for prediction could ingrain that bias, making future decisions based on past (biased) decisions. Being able to assess the reasons as to why a prediction has been made provides another check to ensure fairness.

In 2021, the European Union proposed the AI act (European Commission, 2021), a piece of legislative framework aimed at regulating the development and deployment of AI systems within its member states. This act divides application areas into different risk levels, each with a corresponding level of regulatory oversight. Exploitative systems, such as behavioural manipulation or social scoring, are banned outright. High-risk systems, those which have the potential to affect the rights of citizens, must follow strict rules on transparency and must be assessed before deployment and throughout their life cycle.

However, some analysis of the regulation interprets the ruling as a call for proper regulation and human oversight rather than explicitly calling for transparency-by-design models or the use of XAI techniques (Panigutti et al., 2023).

2.5.3 Style of Explanations for Text

According to Ding et al. (2022), explanations for text and tabular data fall into the following categories: Feature Importance, Rules, Prototype and Counterfactual.

2.5.3.1 Feature Importance

Feature importance is considered the most common type of explanation (Ding et al., 2022; Danilevsky et al., 2020). Each input feature is assigned a value as to how influential it was to the decision. As such, the sign and magnitude characterise how the particular feature affected the instance being explained. A negative sign implies a negative feature importance, in other words, a feature that contributed against the direction of prediction. LIME (Ribeiro et al., 2016a) and SHAP (Lundberg and Lee, 2017a) are prime examples and are explained in detail in the following section.

2.5.3.2 Rule-based Explanations

Rule-based explainability techniques refer to methods that produce human-intelligible rules to describe how input features of a model are logically connected to its output. For example, "if *age* of dog is over 15, then prescribed drug should be..." can be described as an if-then rule. Often, techniques involve the extraction of if-then rules such as these(Wang and Rudin, 2014; van der Waa et al., 2021). Anchors (Ribeiro et al., 2018) represent a rule-based development from the same authors as LIME, with the algorithm producing rules that act as sufficient conditions for a local prediction.

2.5.3.3 Prototypes

In a prototype explanation, the output is given as a similar example or examples from the dataset to provide the user with a knowledge of how other similar entries were treated by the model. Providing the central instance in a cluster also fits the description of a prototype.

2.5.3.4 Counterfactuals

Similarly to prototype explanations, counterfactual explanations also provide a previously labelled example in order to provide a real-world case for the user. However, counterfactual explanations will provide examples with an opposing label, answering what would need to be different about the input in order for the prediction to change. Counterfactual explanations are common as humans as this mirrors the human desire for *contrastive* explanations (Miller, 2018). Examples include Poyiadzi et al. (2020), who ensured that counterfactuals were feasible and actionable, and Mothilal et al. (2020), who generated a diverse *set* of counterfactuals in order to convey the complexity of the task to the user.

2.5.4 SHAP

At its core, SHAP (Lundberg and Lee, 2017a) leverages the concept of Shapley values (Shapley, 1952) to quantify the individual contribution of each feature to a model's prediction. By simulating coalitions of present and absent features and observing the corresponding changes in output, SHAP derives the marginal contributions of each feature to the final prediction (Molnar, 2020). For an original feature of x, a simulated present or absent feature is $x' \subset [0,1]^M$ where M is the maximum number of coalitions. The marginal contribution assigned to each feature i is denoted as ϕ_i . $g(x')$ is the explanation model consisting of the feature importance values multiplied by the corresponding x' value.

$$g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i \tag{2.9}$$

2.5.4.1 Properties of SHAP

SHAP carries three desirable properties that differentiate SHAP from the techniques mentioned in the following section (Section 2.5.6).

The properties are:

1. **Local Accuracy**: Local Accuracy denotes that the prediction of the simplified model for the data point x' must equal the original prediction $f(x)$. This is missing in LIME, where an approximated linear model has no guarantees of bisecting the original data point.
$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i \tag{2.10}$$
2. **Missingness**: SHAP revolves around simulating the presence and absence of features. Missingness states that if the feature is missing in the original model, then it must be given zero feature importance.
$$x'_i = 0 \implies \phi_i = 0 \tag{2.11}$$
3. **Consistency**: If simplified input x'_i increases or stays the same, then its feature importance value ϕ_i should not decrease.

   $f_x(z_0) = f(h_x(z_0))$ and $z_0 \setminus i$ denotes setting $z'_i = 0$. For any two models f and f', if
$$f'_x(z') - f'_x(z' \setminus i) \geq f_x(z') - f_x(z' \setminus i) \tag{2.12}$$
   for all inputs $z' \in \{0,1\}^M$, then
$$\phi_i(f', x) \geq \phi_i(f, x). \tag{2.13}$$

In full, the formula for calculating SHAP values is as follows:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!}[f_x(z') - f_x(z' \setminus i)] \tag{2.14}$$

where $|z'|$ denotes the number of non-zero entries in the vector z', and $z' \subseteq x'$ indicates all z' vectors where the non-zero entries are a subset of the non-zero entries in x'.

2.5.5 Implementing SHAP

In practice, SHAP requires three components to elicit an explanation: a model, an explainer, and a masker. The explainer's role is to dictate which coalitions are to be formed, which calls on the masker to realise these binary coalitions as model-compatible inputs. These inputs are subsequently used by the model to generate a prediction. The choice of explainer and masker, both classes of the SHAP library, depends on the format of the data.

2.5.5.1 Explainers

There exist several variants of the explainer class, such as model-specific SHAP explainer classes designed to elicit speed increases for certain model types or those that trade reduced computation for an approximated version of SHAP.

Here, the focus is on two explainers. The *Permutation* explainer follows the formula laid out in Equation 2.14 and is the default for a model-agnostic approach using tabular datasets. As the name suggests, every possible permutation of feature combinations is trialled in order to get an exact calculation.

When dealing with text inputs or tabular data with many features, calculating the result of every possible coalition can be computationally intensive, especially for long text inputs that may consist of hundreds of tokens. The preferred solution is to use the *Partition* explainer, which organises features into a hierarchy and recursively calculates Shapley values. The resulting values are known as Owen values in game theory. For text, the partition hierarchy is created using a scoring system that clusters neighbouring tokens together, favouring tokens belonging to the same word or not separated by punctuation. When used for tabular features, features are grouped based on their correlation.

For tabular data, the masker uses a background dataset to sample missing features, replacing missing features with a random sample and then integrating over the marginal distribution. For text data, an absent word piece is replaced with a mask token.

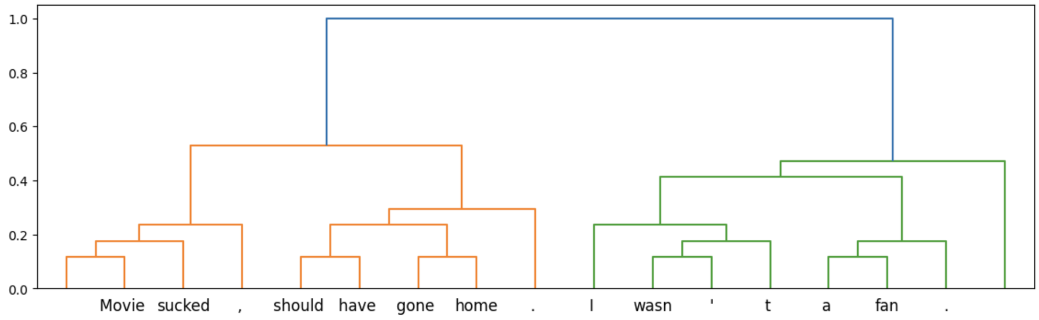


Figure 2.4: A partition tree for the text input 'Movie sucked, should have gone home. I wasn't a fan.'

Figure 2.4 shows an example of the resulting hierarchy, known as a partition tree. The closer to the x-axis, the closer the relationship between the words. One can see that words belonging to the same sentence are grouped more closely, whereas the separate sentences are the most different.

2.5.5.2 Maskers

SHAP hinges upon the simulation of present and absent features in order to make its calculations. Shapley values solved a game-theory problem, where players - either contributing or not - wanted to know how much of the total outcome they were deserving of.

Outside of this scenario, it is not always straightforward to set what it means for a feature to be absent. Maskers, whose task is to realise these simulated absences, work in different ways depending on the category of input: here, this subsection discusses how masking functions for text features and tabular features.

For text, the process is quite simple: each absent text feature - each word piece is treated as a separate feature - is replaced with a [MASK] token. Note that words can be made of more than one word-piece and that punctuation is also to be accounted for. 1s and 0s represent the desired coalition of present and absent features.

Masterpiece of a film .

1 1 1 1 0 1

Masterpiece of a [MASK] .

As for tabular features, it is slightly more complex as there is no guarantee that an empty or NULL value has any meaning for a particular model; hence a different approach is used.

Manufacturer	Price (£'000s)	Colour
Alfa Romeo	35	Green

0 1 1

Manufacturer	Price (£'000s)	Colour
Renault	35	Green
Ford	35	Green

Here, absent tabular features are replaced value(s) from a background dataset, if absent features are chosen to be replaced by more than one value, then the outcomes for all background replacements are calculated and a mean is taken. The SHAP documents suggest that 1, 10, 100 or 1000 are reasonable choices for the number of samples to take. In this example, 'Alfa Romeo' is replaced by two: 'Renault' and 'Ford'.

2.5.6 Other Explainability Methods

The intricacies of how SHAP works are key in understanding the contributions of Chapters 5 and 6. However, Chapter 4 also revolves around the outputs of explainability methods: feature importance values produced by a selection of techniques are used as the input to a language model. SHAP was covered previously, so here, in this section, the remaining feature importance-based explainability techniques used in Chapter 4 are described.

2.5.6.1 LIME

Proposed by Ribeiro et al. (2016a), LIME constructs a local approximation for each prediction to be explained. The idea of LIME is to build a simple, linear surrogate model in the vicinity of the instance that needs to be explained. Points around the instance are sampled, with closer points having more influence than ones further away, and then a linear model is fit to this newly sampled data. By observing the weights of the new linear model, one can now get a sense of which input features are the most important. Despite their generality, LIME has a relatively high computational cost due to the need to process the whole sample to provide a single explanation (Bodria et al., 2020).

2.5.6.2 Integrated Gradients

Integrated Gradients (Sundararajan et al., 2017a) (IG) is a gradient-based method seeking to trace the path of the model prediction by computing the gradient of the model's prediction with respect to the input features. IG is not a model-agnostic method, although it is flexible: IG can be applied to any differentiable model. Furthermore, it requires no extra adjustments to the original model and no additional approximator model to be created. The first method to simply calculate the gradient was First Derivative Saliency (Simonyan et al., 2013); however, the significant issue with saliency is that it becomes insensitive when predictions saturate due to the ReLU function. Integrated Gradients improved on the original saliency method by considering all points along the straight line path from baseline (zero vector) to input. Gradients are computed at each point and then accumulated.

2.5.6.3 Layerwise Relevance Propagation

Layer-wise Relevance Propagation, or LRP (Bach et al., 2015), is another gradient-based method originally designed for images. The influence of each input feature is calculated by backpropagating through the network, satisfying a layer-wise conservation principle. The goal is for the predicted scalar value to be distributed amongst the input features. This is done for each target class separately. Later, it was developed for use in NLP by Arras et al. (2016), where instead of pixels, the text input is represented by a vector.

2.5.7 Multi-modal Explainability

Multimodal explainability remains an emerging field. This section gives a brief overview on the developments that have been in the text-image domain, which is not a focus of this thesis but is relevant to getting some insight on how multiple modalities are combined together. Chapter 4 will address the gap in the research, namely the lack of options to generate explanations for text-tabular problems.

In the text-image domain, which is not the focus of this thesis, there have been some developments in bringing existing explainability techniques to multi-modal problems. For example, Lyu et al. (2022) proposes DIME (pictured in Figure 2.5), which adapts LIME to work for text and images. The individual contributions of each modality are calculated separately and combined with a derivation of the interaction effect to get an explanation that reflects the behaviour of the multi-modal model.

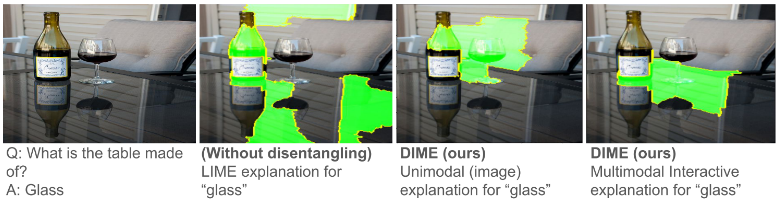

Q: What is the table made of?
A: Glass
(Without disentangling)
LIME explanation for "glass"
DIME (ours)
Unimodal (image) explanation for "glass"
DIME (ours)
Multimodal Interactive explanation for "glass"

Figure 2.5: Lyu et al. (2022) proposes DIME: a multi-modal adaptation of LIME for the text-image domain.

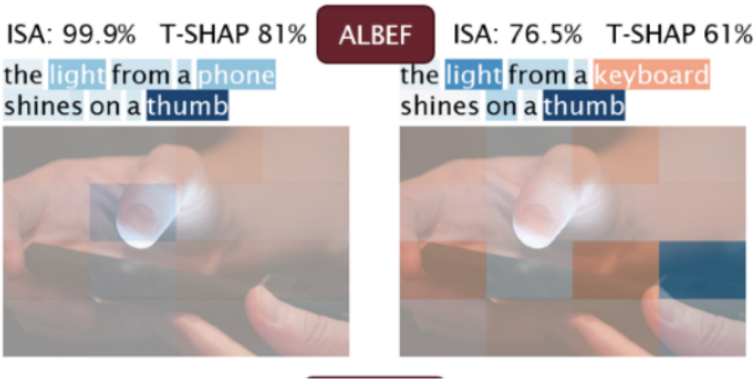

ISA: 99.9% T-SHAP 81%
ALBEF
ISA: 76.5% T-SHAP 61%
the light from a phone shines on a thumb
the light from a keyboard shines on a thumb

Figure 2.6: Parcalabescu and Frank (2023) propose MM-SHAP, adapting SHAP for text-image datasets by blocking images into patches.

Similarly, (Parcalabescu and Frank, 2023) propose MM-SHAP (pictured in Figure 2.6 to provide SHAP explanations for text-image problems. Interestingly, the authors combine SHAP with the accuracy score of the model (typically, explanations are calculated independently of performance) to get an indication of how much each modality is positively contributing to the outcome, with a key focus on assessing the presence of unimodal collapse: a problematic issue for text-image datasets. Images are split into blocks, typically 4x4, but adapted such that text and image inputs have a similar number of features. This will reduce the overall computation and avoid calculating coalitions for each individual pixel, however, the explanations lack a certain precision that is gained by assessing every pixel, as in Lyu et al. (2022).

Although the body of work on explainability is extensive, and adaptations for the text-image domain are growing, there is still a noticeable gap for multi-modal text-tabular explainability. Shi et al. (2021) briefly touches on it in their text-tabular research (although this is not their principal focus). However, it is limited: text features are treated as a whole; SHAP values are reported on a feature level where whole text feature values are substituted for other instances. This suffers from the same lack of precision as Parcalabescu and Frank (2023). To get a true representation of how each tabular feature and each word within text features contribute to the outcome, a new method must be devised. This is the area that will be addressed in Chapter 5.

2.5.8 A Summarisation of the Research Gaps

The deep learning revolution began with the ‘Attention is All You Need’ paper and led to the development of transformer-based large language models such as BERT, DeBERTa and T5. To an exponentially larger extent than the previous generation of models (such as the LSTM), these models could ingest and ‘remember’ vast swathes of information. One disadvantage of these models (that LSTMs share, but not simpler linear models) is their black-box nature, implying an inability to understand how a particular decision had been reached.

Explainability, in general, has experienced a slower rate of development than other research areas, such as those more likely to lead to a bigger, better and more commercially viable model. One such underexplored domain is using large language models themselves to provide additional context and a helpful text-based explanation to make it easier for end users and machine learning practitioners to understand the output of a machine learning decision.

As discussed in Sections 2.5.4 - 2.5.6, for purely text-based data there are a handful of options for producing explanations such as SHAP and LIME from 2017 and 2016 respectively. However, when the data is multi-modal the options become sparser. Section 2.5.7 covered some of the options that have been developed for the text-image domain, such as DIME or MM-SHAP where LIME and SHAP, respectively, were adapted to fit this multi-modal problem. However, for text-tabular machine learning problems, we find a gap in the research, with no method available to generate explanations for text-tabular data. As Chapter 5 will discuss, the one caveat is that one can use SHAP for text if one acts as if the multi-model input is all text, but this leads to errors and is limited to a specific model setup. A general solution remains missing.

Chapter 3

Generating Textual Explanations for Machine Learning Models Performance: A Table-to-Text Task

This chapter proposes a new NLG task aimed at translating a series of evaluation metrics into an analytical narrative describing a model's performance. The goal is to demystify model performance scores for non-experts who may lack a clear understanding of what each metric represents or the implications of the result. Furthermore, this chapter introduces a novel dataset to enable the training of this task. This dataset consists of numerical tables along with corresponding expert-written textual explanations.

This chapter will experiment with this new task and dataset by fine-tuning pre-trained language models (T5 and BART) to generate analytical textual explanations conditioned on the information in the tables. A neural module is proposed, *Metrics Processing Unit* (MPU), to improve the performance of the baselines in terms of correctly verbalising the information in the corresponding table. The evaluation and analysis conducted indicate that exploring pre-trained models for data-to-text generation leads to better generalisation performance and can produce high-quality textual explanations.

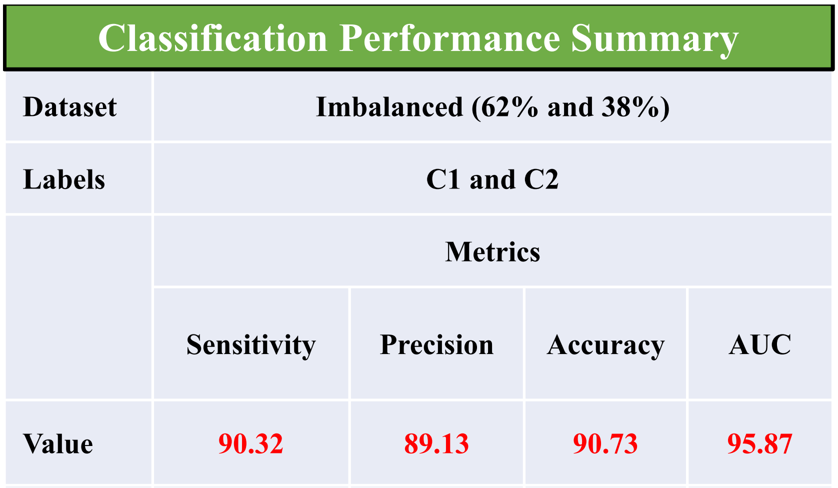
To summarise:

- This chapter proposes a new data-to-text task: starting with metrics that describe a model, produce text that accurately and fluently describes what the metrics are saying
- An expert-written dataset is collected to enable models to do this task
- This chapter proposes a method to linearise the data into text
- This chapter proposes the MPU, an adaption to the model structure to better encode metric information
- The dataset is augmented by permuting the order of the metrics in the linearised data
- A selection of models are trained using linearised data with and without the MPU. Training is done on both the original and permuted datasets.

3.1 Introduction

The performance of trained ML models is widely reported using numerical tables (for example, see the table in Fig. 3.1) and graphs. However, background and domain knowledge are required to make sense of the graphs and tables. Therefore, non-experts will find it more challenging to fully understand the implications of the model's scores across the metrics. In response, a study on training neural models is conducted to generate textual explanations that analytically describe the classification performance of machine learning models.

The generated textual explanation is based on the evaluation metrics' scores achieved, along with information on the underlying dataset (class labels and dataset distribution across the classes) of an arbitrary classification problem. The neural models



Classification Performance Summary

Dataset	Imbalanced (62% and 38%)			
Labels	C1 and C2			
	Metrics			
	Sensitivity	Precision	Accuracy	AUC
Value	90.32	89.13	90.73	95.87

Figure 3.1: A table summarising the classification performance of a classifier.

are trained on table-explanation pairs annotated by computer science experts. Due to the limited size of this dataset, experiments are conducted by fine-tuning the pre-trained language models T5 (Raffel et al., 2020) and BART (Lewis et al., 2020). These pre-trained models treat all text-based language tasks as text-to-text generation; therefore, following Moryossef et al. (2019); Chen et al. (2020a); Suadaa et al. (2021b), the performance summary tables are linearised as flat strings. However, converting structured data to flat strings can result in the loss of important information and relations (Mager et al., 2020; Hoyle et al., 2021; Suadaa et al., 2021b). Therefore, exploring strategies to improve the encoding of the structured data can further improve the quality of the generated output texts. To this end, a neural module is proposed, the MPU, to improve the performance of the pre-trained language models in terms of producing textual explanations and verbalising correctly the information in the corresponding table. The MPU is employed to learn a semantic representation by directly encoding the information about the metrics from the table. The encoder combines the MPU's output representation with the embedding of the linearised representation to generate the contextualised joint representation information passed to the decoder. The contributions of this work are as follows:

- Introducing a new dataset for generating analytical textual explanations describing the performance of classifiers on several machine learning tasks. The textual explanations are written by computer science experts and checked manually to ensure that they accurately reflect or verbalise the information in the corresponding performance table. To the best of our knowledge, this is the first of its kind to focus on explaining the performance of ML models.

- Proposing a neural module, the MPU, which improves the encoding of the information about the metrics, ensuring that the outputs of the pre-trained models accurately verbalise the performance report summarised by the related table.
- Experimenting with state-of-the-art neural models to demonstrate the opportunities and challenges for future research on this table-to-text generation task.

3.2 Proposed Dataset

As stated above, the methodology contributions can be split into two parts: the introduction of a new dataset and developing improvements so as to better encode the information in the table of metrics. This section details the proposed dataset.

3.2.1 Dataset collection

To acquire the dataset for this study, different ML models were trained on 59 classification tasks across different application domains. Across each classification task, five different classifiers were trained. These classification models include random forest, support vector machines, logistic regression and K-nearest neighbour (KNN). For simplicity, only the common classification metrics (accuracy, precision, AUC, recall, specificity, F1-score, and F2-score) were considered.

To collect the dataset, we required annotators with background knowledge of what is meant for each of the metrics. Data quality is important, so therefore computer science experts solicited to provide the analytical textual explanations. In order to be as fair as possible, we approached all of the computer science researchers in the office either studying for a PhD or working as a post-doc. From this group, this summed to ten expert annotators in total. Again in the name of fairness, tasks were randomly assigned to each participant and the same style of questioning was used for all participants. The annotators were all provided with the same style of document to fill in the annotations, either as a pdf or through a web platform. An example of a document used to collect annotations is provided in Figure 3.2. This task does not require the annotators to provide any personal information, however, the annotators were informed of the nature of the task and all ten gave their consent for the answers to be used in this study.

The concept of good performance or a high score is all relative; certain domains require a far higher level of accuracy to be considered successful, medicine being a key example. For this reason, the annotators were also asked to give each of the metric scores on a 3-point scale (High, Moderate, and Low). As part of the annotation collection, annotators were provided with a summary of the task. In this manner, applications of this work can provide their own guide as to what they deem as high, moderate and low. These ratings are used to enrich the metric tables that are passed to the model, providing it with additional context and enhanced numerical reasoning.

Each annotator was provided with the following information:

- A description summarising the objective and the nature of the task at hand
- Definitions of all the metrics used in this example
- Scores of the metrics
- Information on the data split between the classes

They were then instructed to respond to the following:

Table 3.1: Statistics of the model performance narrations dataset

Property	Original	Augmented
Size of train set	725	4529
Size of test set	100	100
Unique words	3548	3548
Min. summary length	35	
Max. summary length	160	
# summaries with 2 sentences	113	579
# summaries with 3 sentences	347	1844
# summaries with >3 sentences	365	2206

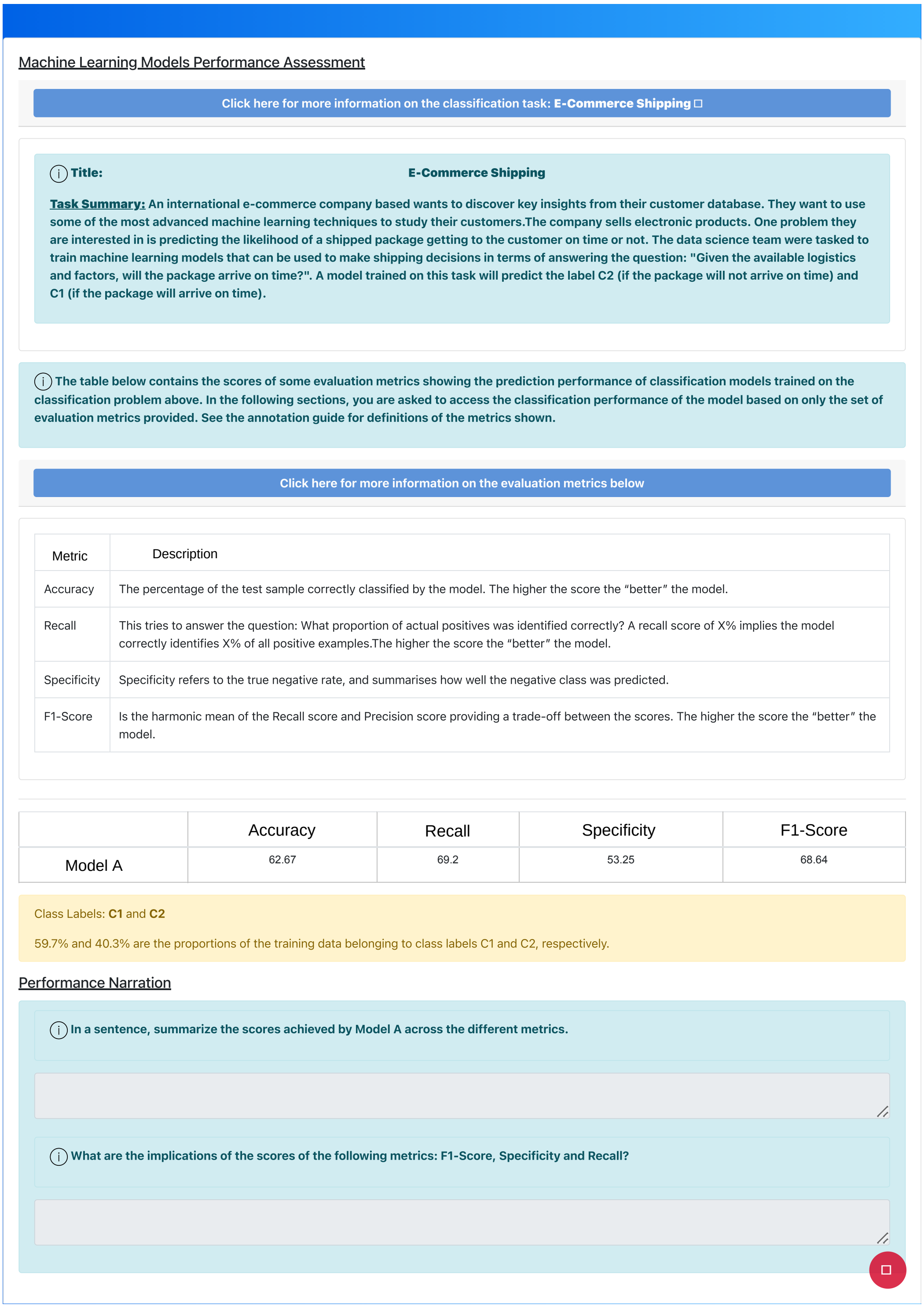
- ‘Provide a summary of the scores achieved by the model across the evaluation metrics.’
- ‘Discuss the overall performance of the model as shown by the values of the evaluation metrics. (Your answer should capture the implications of achieving such scores across the different metrics.)’

3.2.2 Dataset Composition

In total, collected 1010 expert annotations were collected. Each submission was subsequently manually checked for accuracy by comparing it to the corresponding metric table. Out of 1010 submissions, 825 passed this stage and provided a strong, accurate representation of the information in the table. Subsequently, 100 table-explanation pairs were randomly sampled to be used as the test set, with the remaining 725 pairs forming the training set.

3.2.3 Dataset Augmentation

As the dataset is relatively small, augmentation is used to increase the size of the training set. As each metric table has no inherent order, it is possible to increase the size without generating any new instances simply by reordering metrics within the existing instances. When processed, the input will contain the same narratives,



Machine Learning Models Performance Assessment

Click here for more information on the classification task: **E-Commerce Shipping** □

ⓘ **Title:** **E-Commerce Shipping**

Task Summary: An international e-commerce company based wants to discover key insights from their customer database. They want to use some of the most advanced machine learning techniques to study their customers.The company sells electronic products. One problem they are interested in is predicting the likelihood of a shipped package getting to the customer on time or not. The data science team were tasked to train machine learning models that can be used to make shipping decisions in terms of answering the question: "Given the available logistics and factors, will the package arrive on time?". A model trained on this task will predict the label C2 (if the package will not arrive on time) and C1 (if the package will arrive on time).

ⓘ The table below contains the scores of some evaluation metrics showing the prediction performance of classification models trained on the classification problem above. In the following sections, you are asked to access the classification performance of the model based on only the set of evaluation metrics provided. See the annotation guide for definitions of the metrics shown.

Click here for more information on the evaluation metrics below

Metric	Description
Accuracy	The percentage of the test sample correctly classified by the model. The higher the score the "better" the model.
Recall	This tries to answer the question: What proportion of actual positives was identified correctly? A recall score of X% implies the model correctly identifies X% of all positive examples.The higher the score the "better" the model.
Specificity	Specificity refers to the true negative rate, and summarises how well the negative class was predicted.
F1-Score	Is the harmonic mean of the Recall score and Precision score providing a trade-off between the scores. The higher the score the "better" the model.

	Accuracy	Recall	Specificity	F1-Score
Model A	62.67	69.2	53.25	68.64

Class Labels: **C1** and **C2**

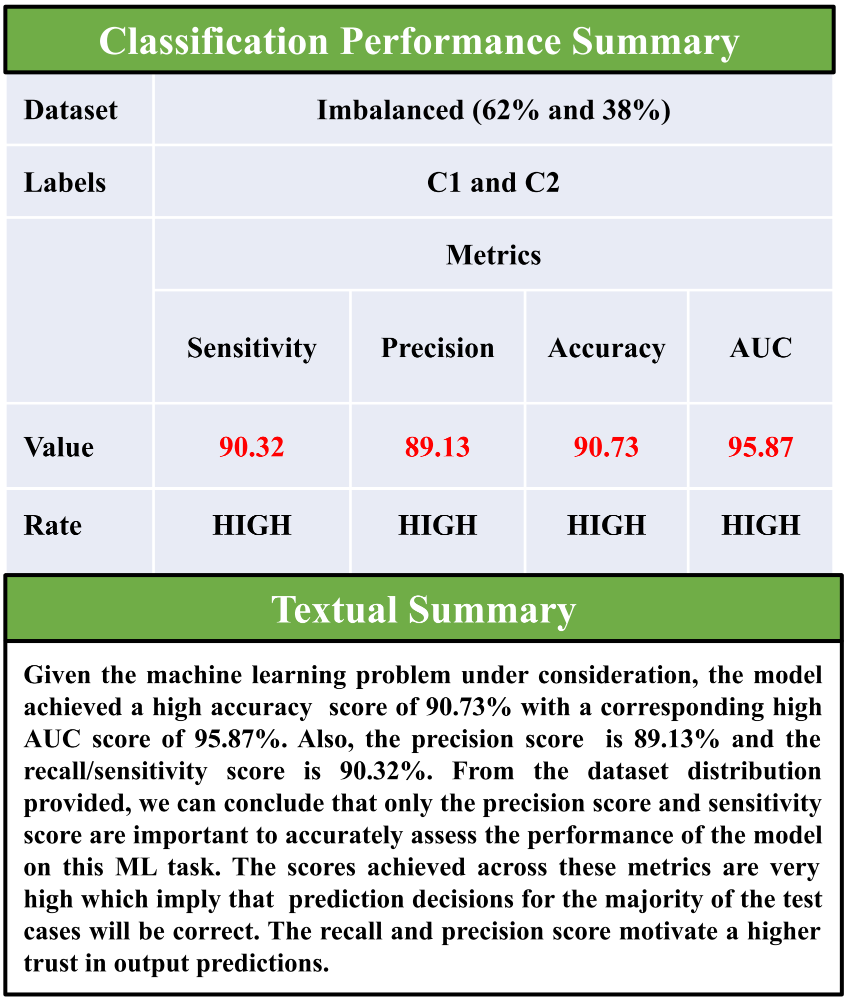
59.7% and 40.3% are the proportions of the training data belonging to class labels C1 and C2, respectively.

Performance Narration

ⓘ In a sentence, summarize the scores achieved by Model A across the different metrics.

ⓘ What are the implications of the scores of the following metrics: F1-Score, Specificity and Recall?

Figure 3.2: An example of the document used to collect annotations



Classification Performance Summary

Dataset	Imbalanced (62% and 38%)			
Labels	C1 and C2			
	Metrics			
	Sensitivity	Precision	Accuracy	AUC
Value	90.32	89.13	90.73	95.87
Rate	HIGH	HIGH	HIGH	HIGH

Textual Summary

Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13% and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high which imply that prediction decisions for the majority of the test cases will be correct. The recall and precision score motivate a higher trust in output predictions.

Figure 3.3: An example of a classification performance report table (containing the score with respect to each metric along with information on the distribution of the underlying data across the two classes: C1 and C2) and the corresponding textual explanation.

but the processed version of the table, as well as the input representation from the proposed neural module, will be different.

In image-based problems, the word *augmentation* is used to describe altering the image in some way, perhaps with a rotation, translation or addition of noise. In text-based problems, it is rarer to have the opportunity to create an augmented dataset. However, by reordering the metrics, or in other words through a *permutation*, we can create an expanded text dataset without adding more data. Note that this thesis will use the terms *augmentation* and *permutation* interchangeably.

The thought behind augmentation was twofold: first, reordering the features will signal to the model that the order of the data does not matter, and it should look for other markers to obtain the right information. Secondly, increasing the size of a relatively small dataset will give the model additional training data, boosting its chances of performing well on the test set. This process increased the training set size from 725 to 4529 table-explanation pairs. Table 3.1 provides some statistics about the dataset.

3.3 Proposed Improvements

3.3.1 Problem Definition

In this work, the input to the NLG models is a numerical table containing the class names, names, values, annotator ratings of each of the metrics, and a dataset distribution indicator. Specifically, the metrics table $T = [C, M, V, R, D]$ is represented as:

- The list of class labels, $C = [C_1, C_2, \cdots, C_k]$
- The list of metric names, $M = [m_1, m_2, \cdots, m_n]$
- The list of metric values, $V = [v_1, v_2, \cdots, v_n]$
- The list of annotator ratings, $R = [r_1, r_2, \cdots, r_n]$
- An indication of whether the dataset was balanced or not, $D \in \{is\_balanced, is\_imbalanced\}$

where k is the number of labels in the example, and n represents the number of metrics in each sample. n_max represents the maximum number of metrics for a sample in the dataset. The target is to generate an analytical textual explanation: a passage of text that should be fluent and factually supported by T.

3.3.2 Linearisation only

Fundamentally, we recognised that using a pre-trained language model was going to be key to producing coherent passages of text. Firstly, a model that is already capable of understanding language is inherently useful. Secondly, the relatively small size of the dataset would make it difficult to learn both the rules of language and the specificities of this particular task without additional fine-tuning. Even with a limited amount of data, fine-tuning pre-trained language models, such as BART and T5, has been shown to produce high-quality texts (Peng et al., 2020; Su et al., 2021; Suadaa et al., 2021b), therefore it was a choice with clear upside. However, BART and T5 were not trained on any table-to-text generation task, and as such, this data-to-text task is reframed as a text-to-text generation problem, following Chen et al. (2020a); Suadaa et al. (2021b); Hoyle et al. (2021).

Before introducing the proposed improvements, a baseline is outlined. Whereas in the following section, adjustments are made to the model structure, here the models are unchanged; the only task is to form the metric table into a text input. To form the input table into text, following Moryossef et al. (2019); Chen et al. (2020a); Suadaa et al. (2021b), the input table is linearised into a flat string. In this chapter, the linearisation is performed by concatenating metric information, class labels, and the dataset distribution based on the following template:

<MetricsInfo> m_1 | VALUE_r_1 | v_1 && m_2 | VALUE_r_2 | v_2 && $\cdots$&& m_n | VALUE_r_n | v_n <|section-sep|> <TaskDec> ml_task | dataset_dist | D && ml_task | class_labels | $C_1, C_2, \cdots,$ and C_k <|section-sep|> <|table2text|>

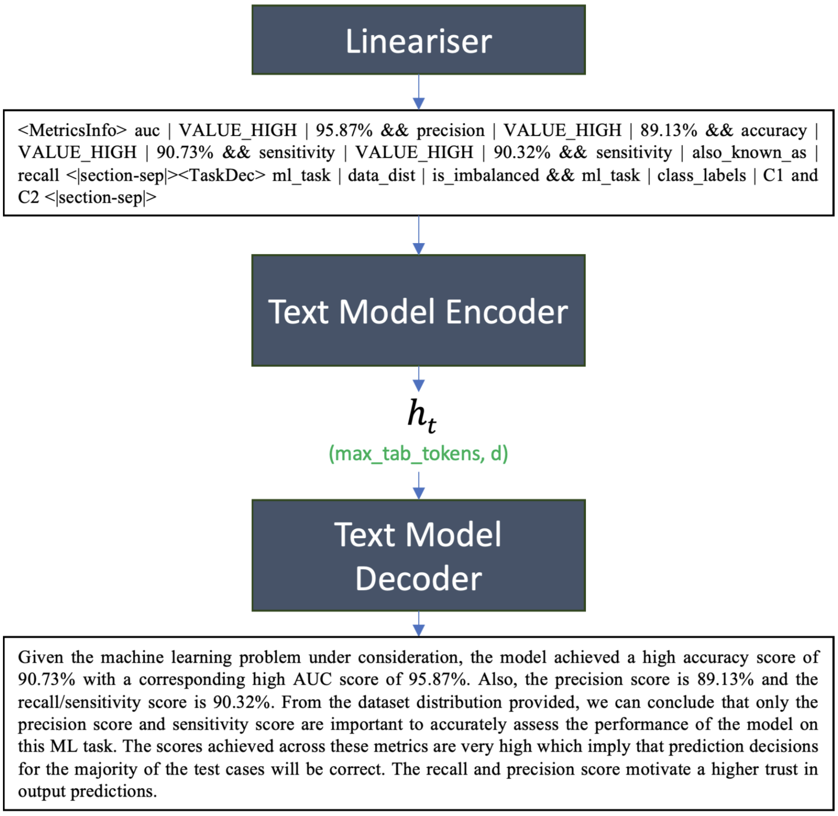
Red text indicates values replaced by those in the table, whereas black text indicates part of the template. An example is given below; Figure 3.3 shows the numerical table pre-linearisation.

<MetricsInfo> auc | VALUE_HIGH | 95.87% && precision | VALUE_HIGH | 89.13% && accuracy | VALUE_HIGH | 90.73% && sensitivity | VALUE_HIGH | 90.32% <|section-sep|> <TaskDec> ml_task | dataset_dist | is_imbalanced && ml_task | class_labels | C_1 and C_2 <|section-sep|> <|table2text|>

Once linearised into strings, LLMs can be fine-tuned with this dataset. During training, each string is tokenised and fed into an LLM. The LLM encoder then converts the tokens into a vector representation of the string, which is then passed to the decoder, which in turn will iteratively generate output tokens. This is the typical manner in which LLMs are fine-tuned; no further alterations are made to the models for these baseline experiments. To make clear the difference between this experiment and the proposed changes from the proceeding section, the process structure is illustrated in Figure 3.4.

3.3.3 Metrics Processing Unit

One drawback of using only the linearised data input is that it can fail to capture information and relations within structured data (Suadaa et al., 2021b; Mager et al., 2020; Hoyle et al., 2021). In this task the goal is to provide textual explanations that accurately represent the information in the metric table, therefore strategies are explored to improve data representation so as to give the text model a method to more effectively encode the numerical information. To this end, we extend the linearised input with a semantic representation of the metric table generated by a custom neural network, which is referred to as the MPU. The MPU, as shown in Fig. 3.5, comprises two main parts: the initial embedding of the metrics, values and ratings, followed by a comprehensive combining of the three initial embeddings, which results in a single consolidated representation of the metrics, values and ratings.



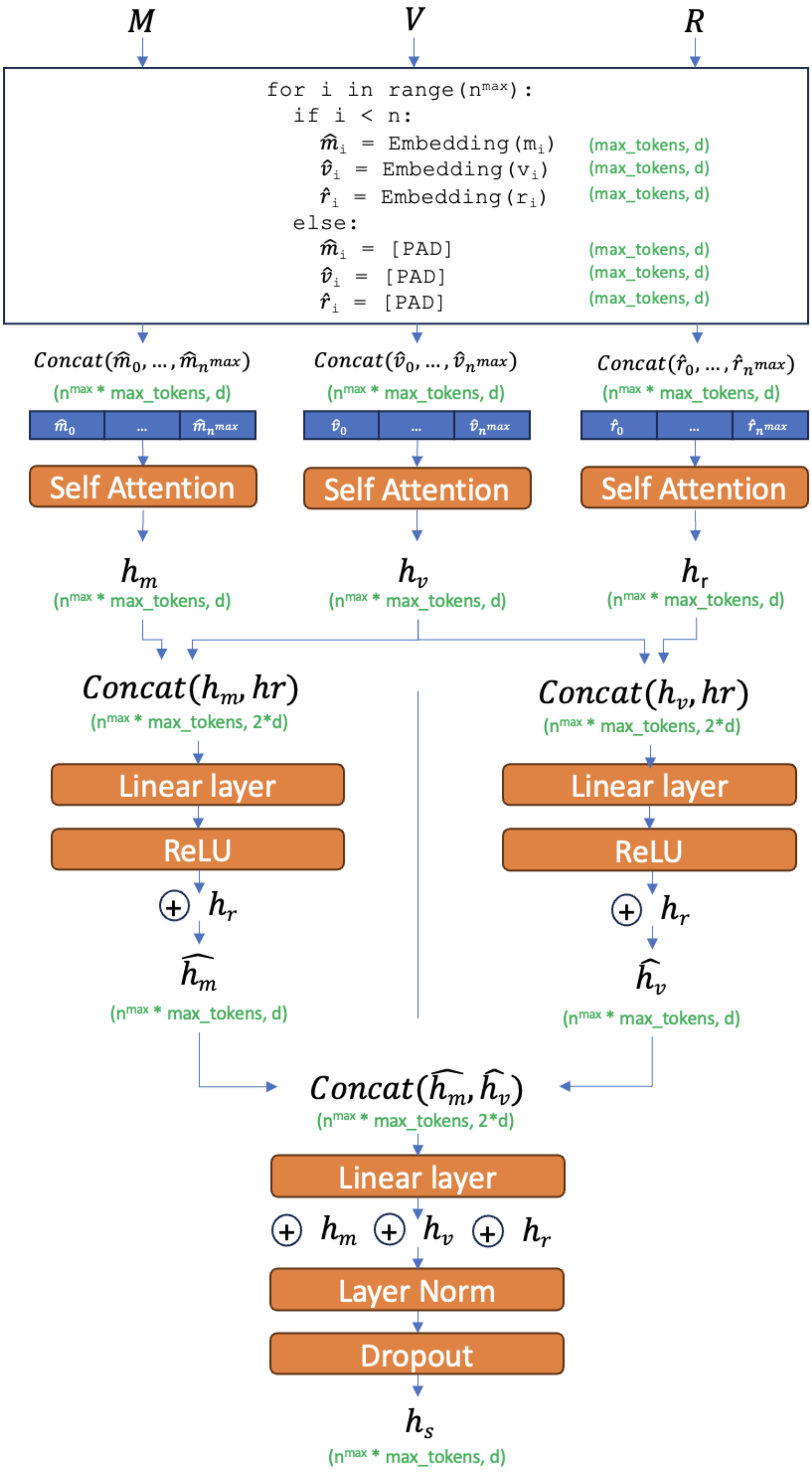
<MetricsInfo> auc | VALUE_HIGH | 95.87% && precision | VALUE_HIGH | 89.13% && accuracy | VALUE_HIGH | 90.73% && sensitivity | VALUE_HIGH | 90.32% && sensitivity | also_known_as | recall <|section-sep|><TaskDec> ml_task | data_dist | is_imbalanced && ml_task | class_labels | C1 and C2 <|section-sep|>

Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13% and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high which imply that prediction decisions for the majority of the test cases will be correct. The recall and precision score motivate a higher trust in output predictions.

Figure 3.4: An illustration of the pipeline without the MPU. A numerical explanation is linearised; it passes through the encoder-decoder of a text model, finishing with the desired text output. Green text indicates the dimensions of the object.

Scores Embedding Unit

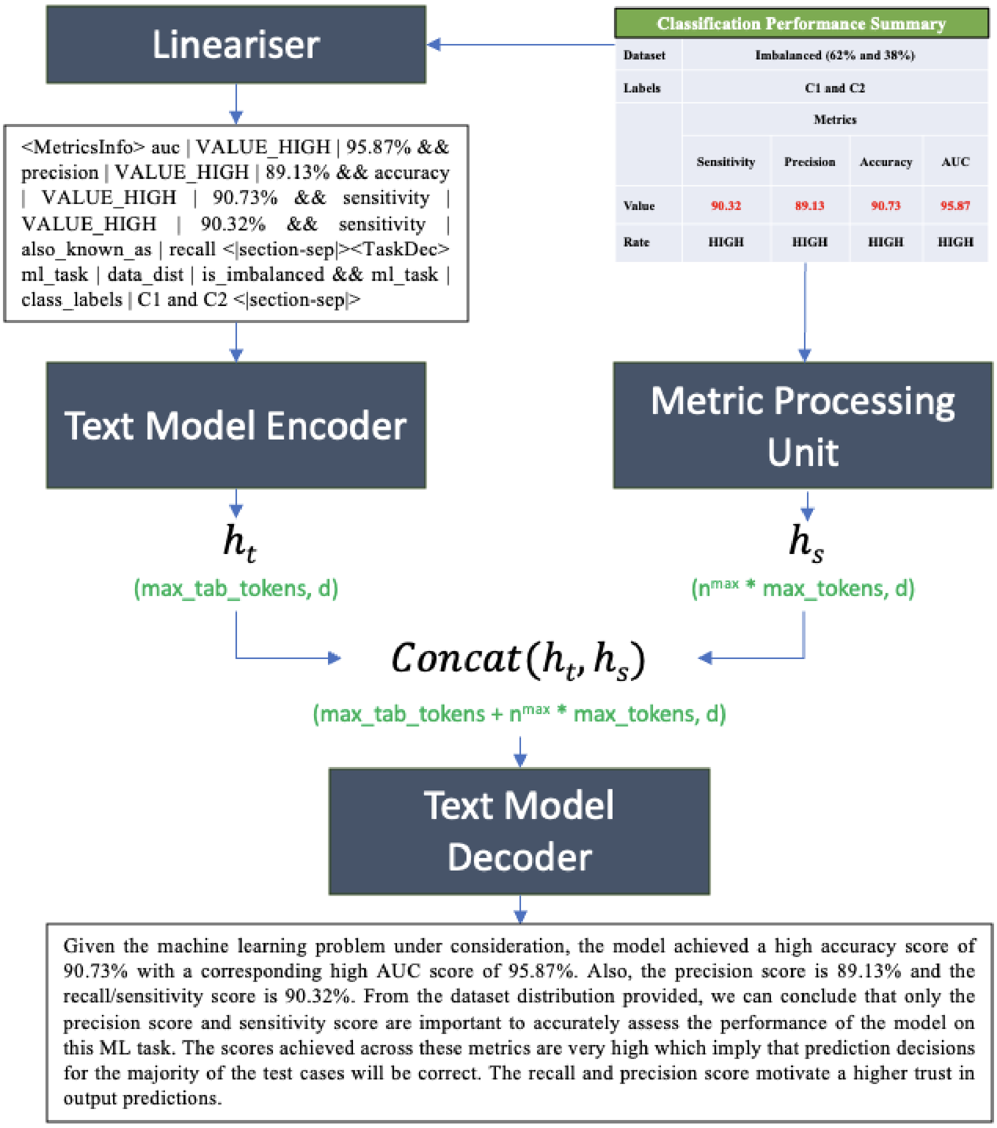
For each metric $m \in M$, we used the text model encoder to generate an embedding $\hat{m_i}$. Not all entries will have the same number of tokens, so each embedding here is padded to max_tokens=8. Each $\hat{m_i}$ is then concatenated together to form $\hat{M}$. The maximum number of scores for a single instance in the dataset is 8; this work will call this n_{max}. So as each is the same length, $\hat{M}$ is padded up to a maximum length of $n_{max} * max\_tokens$ so that each for each instance $\hat{M}$ is of shape $(n_{max} * max\_tokens, d)$ where d is the dimension of the text model's embedding size. Finally, self-attention is applied to $\hat{M}$ to get our hidden representation of the metrics h_m. The same steps are repeated to get hidden representations of an instance's values, h_v, and its ratings, h_r.



```
for i in range($n^{max}$):
  if i < n:
    $\hat{m}_i$ = Embedding($m_i$)      (max_tokens, d)
    $\hat{v}_i$ = Embedding($v_i$)      (max_tokens, d)
    $\hat{r}_i$ = Embedding($r_i$)      (max_tokens, d)
  else:
    $\hat{m}_i$ = [PAD]                 (max_tokens, d)
    $\hat{v}_i$ = [PAD]                 (max_tokens, d)
    $\hat{r}_i$ = [PAD]                 (max_tokens, d)
```

M
V
R
$Concat(\hat{m}_0, \dots, \hat{m}_{n^{max}})$
(n^{max} * max_tokens, d)
$Concat(\hat{v}_0, \dots, \hat{v}_{n^{max}})$
(n^{max} * max_tokens, d)
$Concat(\hat{r}_0, \dots, \hat{r}_{n^{max}})$
(n^{max} * max_tokens, d)
$\hat{m}_0$ ... $\hat{m}_{n^{max}}$
$\hat{v}_0$ ... $\hat{v}_{n^{max}}$
$\hat{r}_0$ ... $\hat{r}_{n^{max}}$
Self Attention
Self Attention
Self Attention
h_m
(n^{max} * max_tokens, d)
h_v
(n^{max} * max_tokens, d)
h_r
(n^{max} * max_tokens, d)
$Concat(h_m, hr)$
(n^{max} * max_tokens, 2*d)
$Concat(h_v, hr)$
(n^{max} * max_tokens, 2*d)
Linear layer
ReLU
$\oplus$ h_r
$\widehat{h_m}$
(n^{max} * max_tokens, d)
Linear layer
ReLU
$\oplus$ h_r
$\widehat{h_v}$
(n^{max} * max_tokens, d)
$Concat(\widehat{h_m}, \widehat{h_v})$
(n^{max} * max_tokens, 2*d)
Linear layer
$\oplus$ h_m $\oplus$ h_v $\oplus$ h_r
Layer Norm
Dropout
h_s
(n^{max} * max_tokens, d)

Figure 3.5: Architecture of the MPU employed to learn to h^s from the list of metric names M, metric values V and annotator ratings R. Green text indicates the dimensions of the object.



Classification Performance Summary				
Dataset	Imbalanced (62% and 38%)			
Labels	C1 and C2			
	Metrics			
	Sensitivity	Precision	Accuracy	AUC
Value	90.32	89.13	90.73	95.87
Rate	HIGH	HIGH	HIGH	HIGH

<MetricsInfo> auc | VALUE_HIGH | 95.87% && precision | VALUE_HIGH | 89.13% && accuracy | VALUE_HIGH | 90.73% && sensitivity | VALUE_HIGH | 90.32% && sensitivity | also_known_as | recall <|section-sep|><TaskDec> ml_task | data_dist | is_imbalanced && ml_task | class_labels | C1 and C2 <|section-sep|>

Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13% and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high which imply that prediction decisions for the majority of the test cases will be correct. The recall and precision score motivate a higher trust in output predictions.

Figure 3.6: An illustration of the table-to-text neural generator trained to produce the textual explanation based on the encoded linearised table h^t and metrics-ratings-values semantic representation, h^s, learned by the MPU.

Interaction Module

This module combines the representations of the metrics, values and ratings (h_m, h_v and h_r respectively) into a single representation of the scores, h_s. It concatenates h_m and h_r and passes them through a linear layer and the ReLU activation function, before adding h_r to get a conjoined metric and rating representation, which will be called $\hat{h}_m$. In parallel, the equivalent is done with h_v and h_r, concatenating the two, passing them through a linear layer and ReLU and adding h_r to get a conjoined value and rating representation, which is called $\hat{h}_r$. To join $\hat{h}_m$ and $\hat{h}_r$, the two are concatenated, passed through another linear layer and all three representations h_m, h_v and h_r are added. Finally, regularisation is added in the form of Layer Norm and Dropout before arriving at the final representation of the scores: h_s. To be compatible with the text encoder and decoder representations, the shape of h_s is ($n^{max}*$ *max_tokens*, d).

Entire Process

Figure 3.6 shows how the whole process fits together. The numerical information is linearised - as per the template detailed above - and passed through the encoder of the text model to get a representation of the table, h_t. h_t is then concatenated with the output of the MPU, h_s, and passed into the text model decoder. The text model decoder will then output the text.

3.4 Results

To test the impact of the proposed improvements, this chapter experimented with variants of pre-trained T5 and BART language models, namely T5-small, T5-base, T5-large, BART-base, and BART-large. Each variant of T5 and BART has the same model structure but differs in the number of parameters.

This chapter experimented with each LLM four times: twice using the MPU (once using the original dataset, once using the augmented version), and twice without the

MPU. For each experiment, the model is trained five times with different random seeds, and the generation performance of the models is reported, based on the average scores for each evaluation metric. For each model variant, '+MPU' refers to models trained with linearisation and the MPU and the absence of '+MPU' refers to models trained simply with linearisation.

3.4.1 Automatic Evaluation

This section presents the automated evaluation of the trained models. Table 3.2 and Table 3.3 show the scores achieved by the models across the selected evaluation metrics: BLEU, METEOR, PARENT, and BLEURT, where Table 3.2 is holding the scores of models trained on the original dataset and Table 3.3 with those trained on the augmented dataset.

Permuted vs Original

In general, training on the augmented dataset led to improved scores. For example, while BART-large scored 28.09 BLEU and 35.55 METEOR when trained on the original dataset, it scored 45.57 BLEU and 46.41 METEOR when trained on the permuted dataset. In fact, training on the permuted dataset leads to increased BLEU and METEOR scores for every model. For the PARENT metric, permutation was only better in half the experiments, with all *large* models doing worse on this metric. Furthermore, for BLEURT, training on the permuted dataset only lead to an improved score in 3 of the 10 experiments, with all *large* models and all but one *base* models showing a better score on the original dataset.

Effect of Metrics Processing Unit

According to Table 3.3, augmenting the linearised representation with the semantic representations of the metrics information in the table further improves the generation performance of the underlying models. The T5-small benefited most among all the models. Specifically, there is a +2.75, +3.94, +1.92, and +0.6 increase in the BLEU, METEOR, PARENT and BLEURT, respectively. Furthermore, T5-small+MPU is shown to have the best match to both the reference text and the source table, according to PARENT. It outperforms the next best models, BART-base+MPU and BART-large+MPU, by +0.77 PARENT and +0.78 PARENT, respectively. Furthermore, despite having a poor surface-level match to the reference texts according to BLEU and METEOR, it achieves the best BLEURT score, meaning its outputs are fluent and semantically equivalent to the reference texts.

Table 3.2: Evaluation of generation performance of the neural models on the original dataset (725 data-text training pairs). '+ MPU' detonates training the variant of the pre-trained baselines with the MPU.

	BLEU	METEOR	PARENT	BLEURT
T5-small	26.00±0.81	32.64±0.66	30.70±0.51	53.12±0.34
T5-small + MPU	26.70±1.12	33.34±1.14	29.39±1.24	53.69±0.72
T5-base	32.83±1.70	35.81±1.36	31.70±0.80	55.51±0.47
T5-base + MPU	32.59±3.60	36.80±1.53	30.36±1.80	55.14±0.36
T5-large	**35.11±2.93**	**38.38±1.69**	32.73±2.01	55.67±0.51
T5-large + MPU	32.27±2.40	36.69±1.11	32.58±2.61	55.84±1.04
BART-base	26.91±3.76	34.90±1.34	31.46±2.33	56.30±0.42
BART-base + MPU	28.79±2.84	35.59±0.93	29.71±3.10	55.13±0.61
BART-large	28.03±2.64	35.55±2.86	32.90±2.39	54.40±1.54
BART-large + MPU	30.13±1.80	38.15±1.93	**34.68±1.10**	**56.48±0.23**

Table 3.3: Evaluation of generation performance of the neural models on the permuted dataset. '+ MPU' detonates training the variant of the pre-trained baselines with the MPU. The models are fine-tuned with five different random seeds, and the scores are based on the average and standard deviation.

Model	BLEU	METEOR	PARENT	BLEURT
T5-small	37.83±2.25	39.46±1.83	32.12±1.16	55.52±0.22
T5-small + MPU	40.58±1.95	43.40±1.05	**34.04±0.65**	**56.15±0.40**
T5-base	46.31±0.89	47.45±0.35	30.99±0.70	54.09±0.30
T5-base + MPU	45.46±0.60	47.75±0.40	31.33±0.67	54.46±0.19
T5-large	45.70±0.73	47.11±0.31	31.08±0.63	54.53±0.20
T5-large + MPU	46.03±0.75	47.53±0.41	31.58±0.73	54.63±0.42
BART-base	44.83±1.03	**47.83±0.22**	32.79±0.86	54.71±0.16
BART-base + MPU	45.75±0.84	**47.83±0.64**	33.26±0.96	55.26±0.31
BART-large	45.57±1.96	46.41±1.44	32.29±2.12	51.29±1.76
BART-large + MPU	**46.98±0.73**	47.52±1.50	33.27±0.6	51.25±1.67

Table 3.4

(a) Example of the model performance narrations generated by the fine-tuned T5 and BART models based on the table data (converted into flat-string). Sentences and phrases conveying correct information according to the related table are highlighted in blue, while incorrect ones are marked in red.

Model	Generated
Table	<MetricsInfo>auc | VALUE_HIGH | 95.87% && precision | VALUE_HIGH | 89.13% && accuracy | VALUE_HIGH | 90.73% && sensitivity | VALUE_HIGH | 90.32% && sensitivity | also_known_as | recall <|section-sep|><TaskDec>ml_task | data_dist | is_imbalanced && ml_task | class_labels | C1 and C2 <|section-sep|> <|table2text|>
Reference	Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13%, and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high, which implies that prediction decisions for the majority of the test cases will be correct. The recall and precision scores motivate a higher trust in output predictions.
T5-small	The performance of the classifier on this binary classification task as evaluated based on the precision, AUC, accuracy, and sensitivity achieved the scores 89.13%, 90.32%, 95.87%, and 90.73% respectively. These scores support the conclusion that this model will be highly effective at correctly classifying most test cases/samples with only a few misclassifications. Furthermore, the confidence in predictions related to the label C2 is very high.
T5-small +MPU	The algorithm's ability to correctly classify test samples as either C1 or C2 was assessed based on the metrics accuracy, AUC, precision, and sensitivity. It scored 90.73%, 95.87%, 89.13% and 90.32%, respectively. As shown in the table, we can confirm that the algorithm has a very high classification performance and will be very effective at correctly labelling most test cases drawn from any of the different classes.
T5-base	On this imbalanced classification task, the trained model scored 90.32% (recall), 90.73% (accuracy), and 95.87% (AUC). From the precision score, it is obvious that the model has a moderately high false-positive rate hence the confidence in predictions related to the C2 label is very high. However, based on the sensitivity and precision scores, we can see that some examples belonging to C1 are likely to be misclassified as C2. This is further supported by the high accuracy and AUC scores.
T5-base +MPU	The classifier scored close to perfect scores across all the metrics (i.e. Precision, AUC, Accuracy and Sensitivity). From the results table, we can see that it scored 90.73% (accuracy), 90.32% (sensitivity), and 95.87% (AUC). Surprisingly, these scores were achieved even though the dataset was imbalanced. With such high scores for precision and sensitivity, the model is shown to have a lower misclassification error rate. Overall, this model has relatively high classification performance, and hence will struggle to correctly identify the labels for only a few test cases belonging to the different classes.

(b) Example of the model performance narrations generated by the fine-tuned T5 and BART models based on the table data (converted into flat-string). Sentences and phrases conveying correct information according to the related table are highlighted in blue, while incorrect ones are marked in red.

Model	Generated
Table	<MetricsInfo>auc | VALUE_HIGH | 95.87% && precision | VALUE_HIGH | 89.13% && accuracy | VALUE_HIGH | 90.73% && sensitivity | VALUE_HIGH | 90.32% && sensitivity | also_known_as | recall <|section-sep|><TaskDec>ml_task | data_dist | is_imbalanced && ml_task | class_labels | C1 and C2 <|section-sep|> <|table2text|>
Reference	Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13%, and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high, which implies that prediction decisions for the majority of the test cases will be correct. The recall and precision scores motivate a higher trust in output predictions.
T5-large	Evaluating the performance of the model on this classification task produced the scores: 95.87% for AUC, 90.73% for accuracy, 89.13% precision, and 90.32% for sensitivity(recall). From the recall and precision scores, we can see that the classification algorithm is very effective at correctly classifying most unseen test cases or samples with onlya small margin of error (the misclassification error rate is 9.27%). The very high accuracy coupled with the very low precision score demonstrates its capability to correctly identify the correct class labels for several test instances.
T5-large +MPU	The classifier scored close to perfect scores across all the metrics (i.e. Precision, AUC, Accuracy and Sensitivity). From the results table, we can see that it scored 89.13%(Precision), 90.32% (sensitivity), 95.87% (AUC), and 90.73%(Accuracy). Since the dataset was imbalanced, it would be wise to analyze prediction performance based on the balance between the recall and precision scores. The precision and sensitivity scores show how good the model is at correctly recognising the observations under the different classes (C1 and C2). In summary, the models are likely to have a lower misclassification error.
BART-base	The classification performance scores achieved by the model on this binary classification task are as follows: (1) AUC score of 95.87, (2) Accuracy equal to 90.73%, (3) Precision score equal 89.13%, and (4) Sensitivity (sometimes referred to as the recall) score is 90.32%. These scores across the different metrics suggest that this model is very effective and can accurately identify the true labels for several test cases/samples with a small margin of error (actually, the misclassification error is 9.27%)
BART-base+MPU	The scores achieved by the AI algorithm on this binary classification task are as follows (1) AUC score of 95.87%, (2) Accuracy equal to 90.73%, and (3) Precision score equal 89.13%. These scores across the different metrics suggest that this model is very effective and can accurately identify the true labels for the majority of the cases with a small margin of error. Furthermore, the precision and recall scores indicate that the likelihood of misclassifying C1 cases as C2 is very marginal (that is, it has a very low false-positive rate).

(c) Example of the model performance narrations generated by the fine-tuned T5 and BART models based on the table data (converted into flat-string). Sentences and phrases conveying correct information according to the related table are highlighted in blue, while incorrect ones are marked in red.

Model	Generated
Table	<MetricsInfo>auc | VALUE_HIGH | 95.87% && precision | VALUE_HIGH | 89.13% && accuracy | VALUE_HIGH | 90.73% && sensitivity | VALUE_HIGH | 90.32% && sensitivity | also_known_as | recall <|section-sep|><TaskDec>ml_task | data_dist | is_imbalanced && ml_task | class_labels | C1 and C2 <|section-sep|> <|table2text|>
Reference	Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13%, and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high, which implies that prediction decisions for the majority of the test cases will be correct. The recall and precision scores motivate a higher trust in output predictions.
BART-large	The performance of the model on this binary classification task, as evaluated based on the precision, AUC, accuracy, and sensitivity, scored 89.13%, 95.87%, 90.73%, and 90.32% respectively, implying that it is a very effective model. These scores indicate that the likelihood of misclassifying test samples is very marginal. However, the scores were expected the dataset was perfectly balanced between the two class labels C1 and C2.
BART-large+MPU	On this imbalanced classification task, the trained model reached an AUC score of 95.87, an accuracy of 90.73, with a precision and sensitivity scores equal to 89.13 and 90.32, respectively. These results/scores are very impressive as one can conclude that this model is almost perfect with higher confidence in its prediction decisions. In summary, only a small number of test cases are likely to be misclassified as indicated by the accuracy, sensitivity, and precision.

3.4.2 Quality Analysis

Table 3.4 shows the performance textual explanations generated by the models under consideration based on the table shown in Fig. 3.3. Sentences and phrases conveying correct information according to the related table are highlighted in green, while incorrect ones are marked in red. As shown, the generators are able to produce high-quality classification performance summaries capturing the information presented in the input structured data. However, there were a number of cases where the generators failed to accurately verbalise the content of the related performance metric table. The errors are mainly from the models trained without MPU, and among these models, only BART-base produced a correct verbalisation of the input table. The summary from T5-small is mostly valid; however, the metrics (AUC, accuracy and sensitivity) and their corresponding scores are mentioned in the wrong order. The T5-base model made an incorrect assessment of the precision and recall scores when it concluded that the 'false-positive rate' is moderately high. In the case of the T5-large, it stated that the precision is very low even though it was rated 'HIGH'. BART-large produced a wrong statement about the distribution of the dataset between the classes, C1 and C2. Augmenting the linearised representations with the metrics-values-ratings, contextual information from MPU allow the T5 and BART models to generate accurate analytical textual explanations based on the related table.

3.5 Conclusion

This chapter presented a new NLG dataset for generating textual explanations that describe the performance of classification models. Presenting the generated texts along with the numerical tables will allow for a better understanding of the classification performance of ML models. Baselines were trained by fine-tuning state-of-the-art pre-trained models: T5 and BART. Experimental results show the feasibility of utilising these large pre-trained language models to generate fluent and accurate statements based on structured data. However, analysis suggests that neural models, in some instances, produce statements containing wrong information according to the input table. This weakness can be attributed to the direct linearisation of the input tables. To address this problem, this chapter introduced the MPU, which, when combined with the linearised input, produced the best performance across the different T5 and BART variants.

Chapter 4

Natural Language Explanations for Machine Learning Classification Decisions

This chapter addresses the challenge of providing understandable explanations for machine learning classification decisions. To do this, this chapter introduces a dataset of expert-written textual explanations paired with numerical explanations, forming a data-to-text generation task. BART and T5 language models are fine-tuned on this dataset to generate natural language explanations by linearising the information represented by explainable output graphs. Additionally, the numerical explanations are probed more directly by fine-tuning BART and T5 on a question-answer task and achieved an accuracy of 91% with T5.

4.1 Introduction

In recent years, there has been an effort to increase transparency in the decision-making process of black-box models used for predictions and incorporate XAI techniques. This is not least - as discussed in Section 2.5 - to be compliant with the proposed AI act (European Commission, 2021) and avoid model bias. Many types of explanation exist, however in this chapter will focus on four commonly used XAI techniques: Local Interpretable Model-Agnostic Explanations (LIME) (Ribeiro et al., 2016b), SHapley Additive exPlanations (SHAP) (Lundberg and Lee, 2017b), Integrated Gradients (IG) (Sundararajan et al., 2017b), and Layer-wise Relevance Propagation (LRP) (Binder et al., 2016). Although distinct, these techniques all produce feature importance values that quantify the contribution of each feature to the prediction. The details of each method are outlined in Sections 2.5.4 and 2.5.6

Graphs and figures are commonly used to communicate the contributions of each variable used to arrive at a given prediction. These graphs produced by XAI techniques indicate which features are positive (supporting the prediction output), negative (contradicting the prediction output), and neutral (having a negligible influence on the prediction decision). However, for non-experts, it can be challenging to fully understand these figures.

Large, pre-trained language models are trained on a vast text corpus, giving them a broad generalised understanding of language. Fine-tuning these models for specific tasks has been shown to improve their task-specific understanding, even with limited training data (Peng et al., 2020; Su et al., 2021; Suadaa et al., 2021b). Two such language models are T5 (Raffel et al., 2020), and BART (Lewis et al., 2020). T5 is a multitask-trained transformer model trained on several unsupervised and supervised NLP tasks, such as classification, summarising, and translation. BART (Lewis et al., 2020) is a transformer-based denoising autoencoder trained to reconstruct the original text from a corrupted input. A more thorough explanation of these models can be found in Section 2.1.2.

This chapter proposes a new task: given a classifier prediction and a subsequent local-level explanation, produce a narrative that describes the explanation. The narrative should be fluent and factually accurate to provide clarity to the end user when provided alongside a figure. The task is designed to be ambivalent to the choice of explainability technique. The only requirement is that the classifier produces a probability estimation across the classes and that the XAI technique produces a score for each input feature.

To achieve this task, consulted computer science experts with knowledge of explainability were consulted to create a new dataset: TEXtual Explanation Narratives (TEXEN). TEXEN comprises local-level explainability outputs and written narratives that explain in plain text what the numerical explanations are showing. After sifting for quality and factual accuracy, TEXEN contains 496 explanation-text pairs.

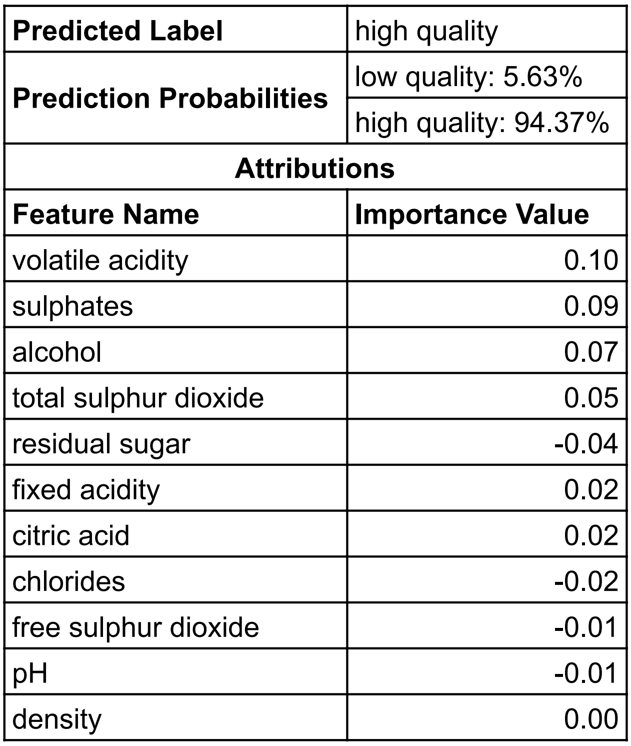
T5 and BART are subsequently fine-tuned on this dataset to generate automatic textual explanations. They are also trained on an augmented version of TEXEN, using the same narratives but reshuffled feature names to artificially increase the size of the training set. Finally, the task is simplified and a question-answer model is trained to respond to questions more directly.

To summarise, the contributions of this chapter are as follows:

- A new dataset for generating textual explanations of a given classification decision is introduced. The textual explanations are written by computer science experts and checked manually to ensure that they appropriately reflect the contribution of input features, as produced by numerical explainability methods. To the best of our knowledge, this study is the first of its kind to focus on generating textual explanations via neural NLG.
- Experimentation and evaluation with state-of-the-art neural pre-trained language models demonstrate the opportunities and challenges for future research on this structured data-to-text generation task.
- Further comparisons are made when simplifying to a more structured question-answer task, with a large synthetically generated training set.

4.2 Background: Numerical Explanation Pipeline

In a typical explainability pipeline, a trained classifier will first make a prediction. To make a local-level explanation, the XAI technique will utilise the prediction and the classifier to yield importance scores for each input feature. For clarity, this chapter refers to the set of features, feature importance scores and prediction probabilities as a *numerical explanation*. An example is shown in Figure 4.1. This is to differentiate it from the object of this task: a *textual explanation* that looks to describe the same set of objects, but in natural language.



Predicted Label	high quality
Prediction Probabilities	low quality: 5.63%
	high quality: 94.37%
Attributions	
Feature Name	**Importance Value**
volatile acidity	0.10
sulphates	0.09
alcohol	0.07
total sulphur dioxide	0.05
residual sugar	-0.04
fixed acidity	0.02
citric acid	0.02
chlorides	-0.02
free sulphur dioxide	-0.01
pH	-0.01
density	0.00

Figure 4.1: An example of a numerical explanation

A typical numerical explanation pipeline is pictured in Figure 4.2. A numerical explanation pipeline aims to explain why a classifier made the decision it did for a particular input data record. This work will only consider explanation methods that produce feature importance scores that can then be shown as a graph. A numerical explanation pipeline consists of the following:

Classifier

Given a test case, a trained classifier generates the classification output decision. This *prediction* is in the form of class labels and their respective predicted probabilities.

Explainer

The explainer's task is to generate *feature importance scores* for each input feature which explains the classification decision of the particular test case, given a trained classifier and the prediction decision. These feature importance scores are then represented as a *graph*, the final output of a typical local-level explanation. In this paper, the explainability techniques used are LIME, SHAP, IG, and LRP.

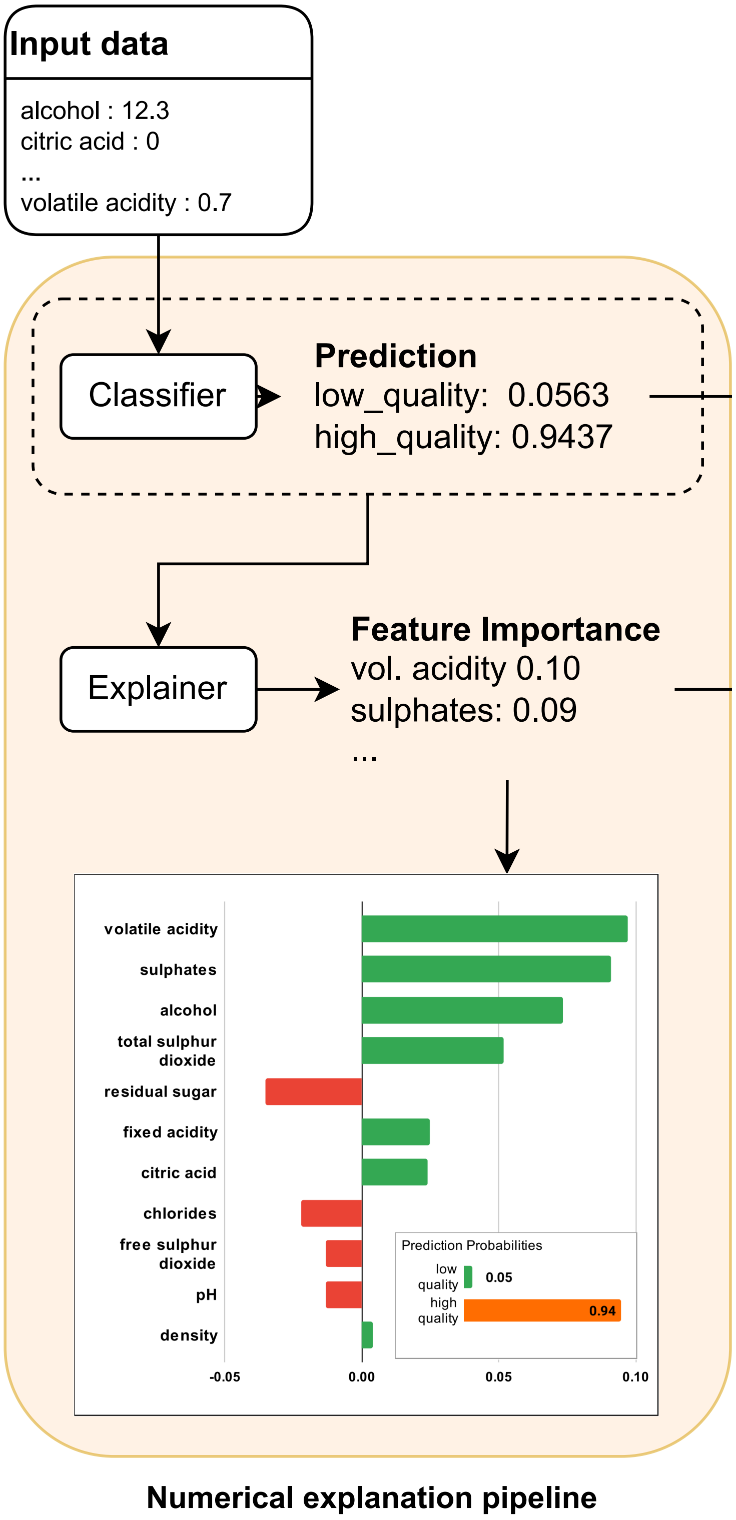
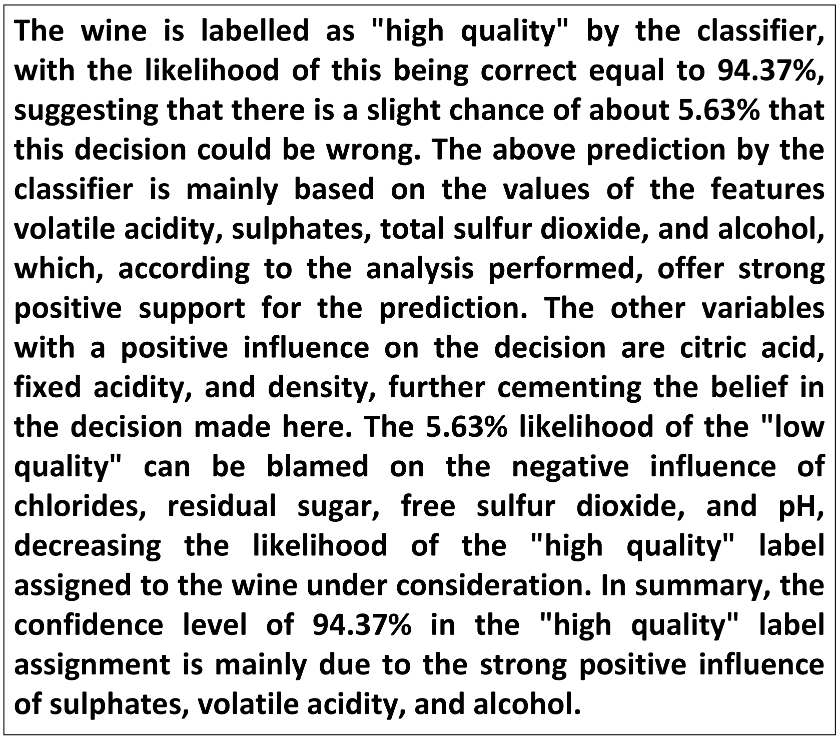


Figure 4.2: A typical numerical explanation pipeline

4.3 Summarisation Task: Proposed Dataset

4.3.1 Dataset Collection

The data collection process involved TEXEN consists of pairs of explanations: one output of a local-level explanation method (a numerical explanation) and written narrative which describes in plain text what the numerical explanation is showing (a textual explanation). An example of a textual narrative is shown in Fig. 4.3).



The wine is labelled as "high quality" by the classifier, with the likelihood of this being correct equal to 94.37%, suggesting that there is a slight chance of about 5.63% that this decision could be wrong. The above prediction by the classifier is mainly based on the values of the features volatile acidity, sulphates, total sulfur dioxide, and alcohol, which, according to the analysis performed, offer strong positive support for the prediction. The other variables with a positive influence on the decision are citric acid, fixed acidity, and density, further cementing the belief in the decision made here. The 5.63% likelihood of the "low quality" can be blamed on the negative influence of chlorides, residual sugar, free sulfur dioxide, and pH, decreasing the likelihood of the "high quality" label assigned to the wine under consideration. In summary, the confidence level of 94.37% in the "high quality" label assignment is mainly due to the strong positive influence of sulphates, volatile acidity, and alcohol.

Figure 4.3: An example of a textual explanation corresponding to Fig. 4.1

First, to collect the numerical explanations, a selection of models were trained on a selection of tasks. Ten different model types were used, including Support Vector Machines, Logistic Regression, Deep Neural Networks, and Random Forests. Using random samples from the test sets, local-level explanations were generated using four XAI techniques: LIME, SHAP, IG, and LRP. These techniques generated numerical scores for each input feature, indicating their relative influence on the classification decision. However, it is necessary to reiterate that these scores do not reflect the accuracy of the classifier but rather provide insight into which features were most important in the decision-making process. Statistics on how the numerical explanations were collected are in Table 4.1.

Table 4.1: Statistics on data used for numerical explanation generation

Property	**Value**
Datasets used:	40
Models used	10
Records per dataset: Mean / S.D.	11.7 / 3.4
Records per model: Mean / S.D.	42.7 / 34.3
Input features per record: Mean / S.D.	18.7 / 15.0

The same methodology of recruiting annotators as Chapter 3 was also used here as once more this was a difficult task which required some expertise in the field. Namely, all researchers in the office either studying a PhD or completing a post-doc were asked to take part in the annotation. By recruiting annotators in this way, we make it as unbiased as we can. In total, eight of these computer science experts took part in the annotation. Similarly to Chapter 3 we did not expect there to be any sort of personal information that could link the annotations to the annotator. Certain annotators may have a particular writing style, say, but we did not consider this to be a significant issue. All were informed of the nature of the research and all gave their consent to be a part of the study.

To collect narratives, eight computer science experts were shown a chart (as in Fig. 4.4) and asked to summarise it in a single text box. These narratives are intended to describe the prediction as a whole. The charts simply represent feature importances in a graphical manner. The feature importance values came from LIME, SHAP, Integrated Gradients and LRP but this task is designed to be ambivalent to the choice of method. The focus is simply on taking a set of feature importance values and translating this into a textual narrative. In the same manner, as the previous chapter, annotators were provided a document, an example of which is shown in Figure 4.5, either through a web platform or directly as a pdf.

In the previous chapter, the same questions were asked to each annotator for each example. However, in this chapter several variations of each question are used so as to elicit a more varied set of responses. A breakdown of the questions used and how many times they appeared in the final dataset is detailed in the following subsection

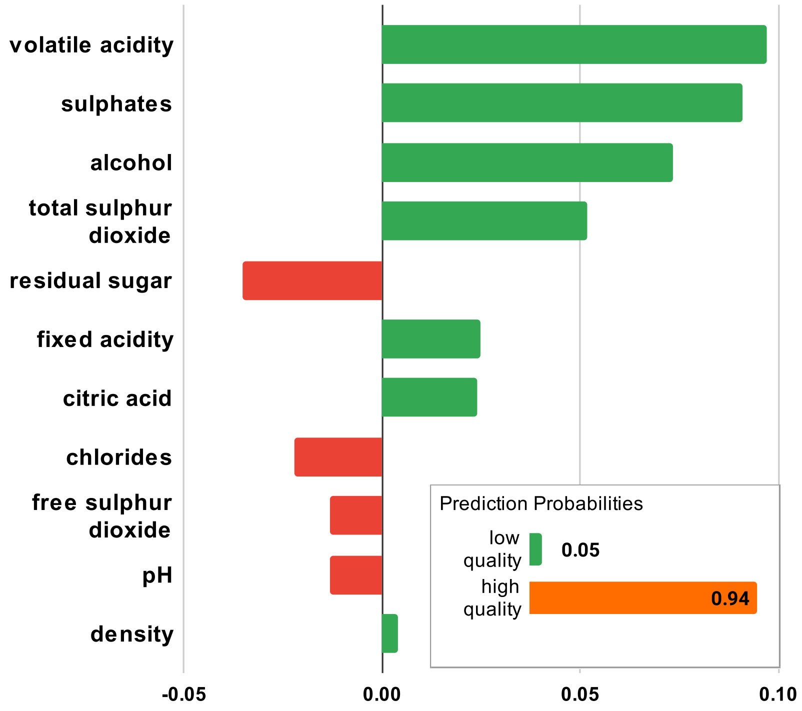
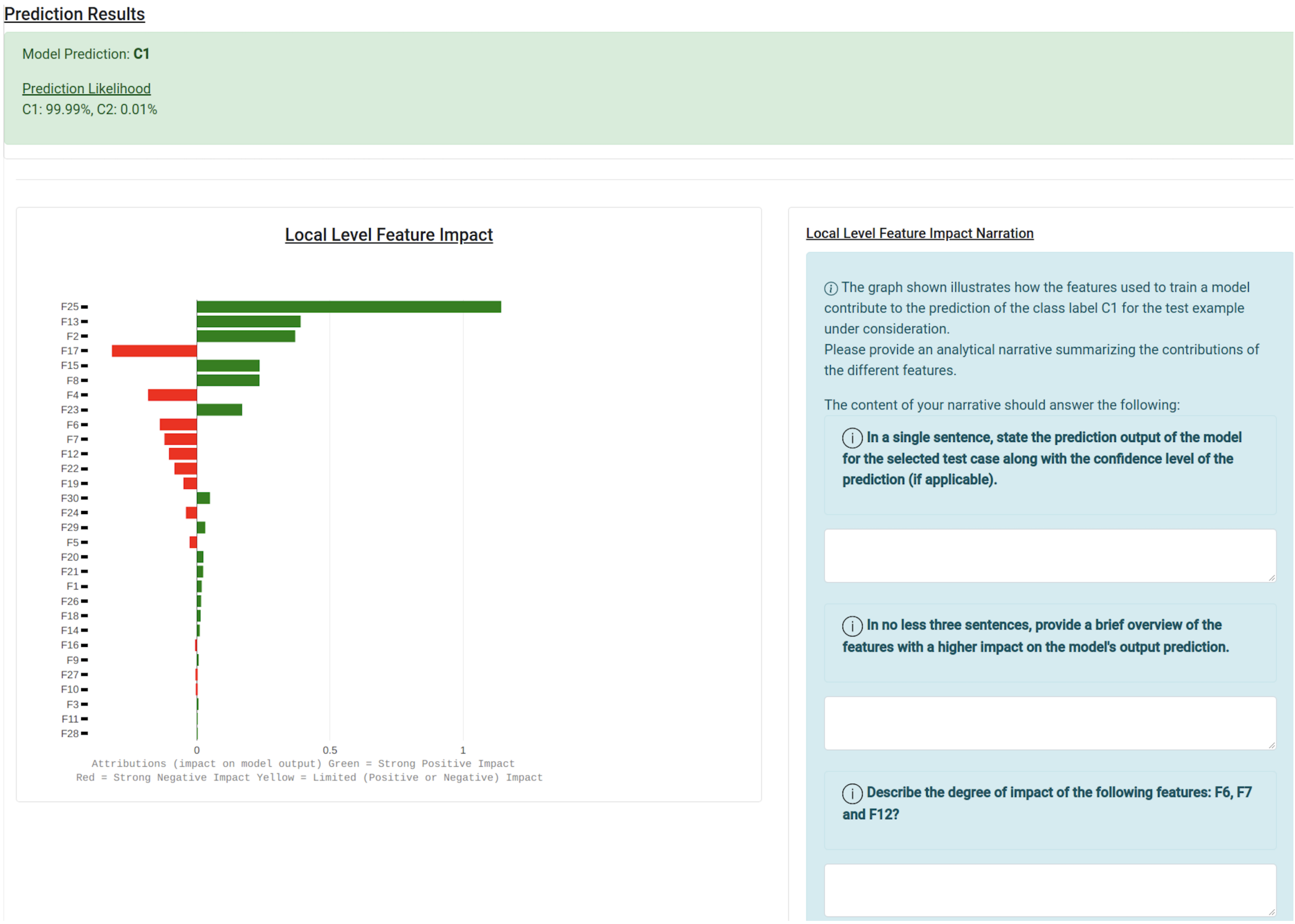


Figure 4.4: An example output graph from LIME, corresponding to Fig. 4.1



Prediction Results

Model Prediction: **C1**

Prediction Likelihood
C1: 99.99%, C2: 0.01%

Local Level Feature Impact

F25
F13
F2
F17
F15
F8
F4
F23
F6
F7
F12
F22
F19
F30
F24
F29
F5
F20
F21
F1
F26
F18
F14
F16
F9
F27
F10
F3
F11
F28
0
0.5
1
Attributions (impact on model output) Green = Strong Positive Impact
Red = Strong Negative Impact Yellow = Limited (Positive or Negative) Impact

Local Level Feature Impact Narration

ⓘ The graph shown illustrates how the features used to train a model contribute to the prediction of the class label C1 for the test example under consideration.
Please provide an analytical narrative summarizing the contributions of the different features.

The content of your narrative should answer the following:

ⓘ **In a single sentence, state the prediction output of the model for the selected test case along with the confidence level of the prediction (if applicable).**

ⓘ **In no less three sentences, provide a brief overview of the features with a higher impact on the model's output prediction.**

ⓘ **Describe the degree of impact of the following features: F6, F7 and F12?**

Figure 4.5: An example of the document provided to annotators to collect the textual explanations.

In order to guide the annotators, they were asked to provide textual explanations that answered the variants of the following questions:

i Summarise the prediction made for the test case under consideration along with the likelihood of the different possible class labels.

ii Summarise the top features influencing the model's decision.

iii Summarise the features with moderate to low influence on the model's decision.

iv Compare the features with positive contributions to those with negative contributions resulting in the classification decision.

4.3.2 Question Breakdown

Specifically, the train set contains exactly these questions.

For 99 cases the format is:

- 'In a single sentence, state the prediction output of the model for the selected test case along with the confidence level of the prediction (if applicable).'
- 'In no less than three sentences, provide a brief overview of the features with a higher impact on the model's output prediction.'
- 'Describe the degree of impact of the following features: [0-4 fts (after first 7-9)]' (3 times there are 0)

For 78 cases the format is:

- 'For this test instance, provide information on the predicted label along with the confidence level of the model's decision.'
- 'Summarise the top features influencing the model's decision along with the respective directions of influence on the prediction.'

- ‘Summarise the direction of influence of the features [the next 3-4 features (after first 2-4)] with moderate impact on the prediction made for this test case.’

For 53 cases the format is:

- ‘Summarise the prediction for the given test example?’
- ‘In two sentences, provide a brief overview of the features with a higher impact on the model’s output prediction.’
- ‘Compare and contrast the impact of the following attributes [3-4 random features] on the model’s prediction of [C1/C2].’
- ‘Summarise the set of features has little to no impact on the prediction?’

For 20 cases the format is:

- ‘Summarise the prediction for the given test example?’
- ‘For this test case, summarise the top features influencing the model’s decision.’
- ‘For these top features, what are the respective directions of influence on the prediction?’
- ‘Provide a statement on the set of features that have limited impact on the prediction of [C1/C2] by the model for the given test example?’

For 39 cases the format is:

- ‘Summarise the prediction made for the test under consideration along with the likelihood of the different possible class labels.’
- ‘Provide a statement summarising the ranking of the features as shown in the feature impact plot.’

- ‘Compare the direction of impact of the features: [2-5 top features].’
- ‘Summarise the direction of influence of the features [the next 3-4 features] with moderate impact on the prediction made for this test case.’
- ‘Provide a statement on the features with the least impact on the prediction made for this test case.’

For 44 cases the format is:

- ‘Provide a statement summarising the prediction made for the test case.’
- ‘For the current test instance, describe the direction of influence of the following features: [2-5 top features]’
- ‘Compare and contrast the impact of the following features [the next 3-4 features] on the model’s prediction of [C1/C2].’
- ‘Describe the degree of impact of the following features: [the next 0-4 features]?’ (usually 4 unless there are not enough features)

For 39 cases the format is:

- ‘Provide a statement summarising the ranking of the features as shown in the feature impact plot.’
- ‘Summarise the direction of influence of the features [2-5 top features] on the prediction made for this test case.’
- ‘Compare the direction of impact of the features: [the next 3-4 features].’
- ‘Describe the degree of impact of the following features:[the next 0-4 features]’

For 3 cases the format is:

- ‘Summarise the prediction made for the test under consideration along with the likelihood of the different possible class labels.’

- ‘Summarise the direction of influence of the variables [2-3 top features] on the prediction made for this test case.’
- ‘Compare the direction of impact of the variables: [the next 3-4 features].’
- ‘Describe the degree of impact of the following variables: [the next 3-4 features]?’

4.3.3 Dataset Composition

We collected 700 textual explanations from the experts, which were manually checked to ensure they correctly articulated the information in the corresponding explanation graph. A majority (469) were shown to accurately capture the information and correctly answer the questions posed to the annotators. Feature and class names were substituted for placeholders and randomised to prevent train-test leakage. The data was divided randomly into training, validation, and test sets (328/47/94). Statistics about the dataset introduced are summarised in Table 4.2.

Table 4.2: Statistics for the local-level textual explanation dataset

Property	**Value**
Size: Train / Validation / Test	328 / 47 / 94
Words per narrative: Mean / S.D.	188 / 47
Unique words	2466

4.3.4 Augmented Dataset

It was hypothesised that the limited training set might impede model performance, therefore this chapter proposes a new augmented training set constructed from the original numerical explanation narrative pairs and substituting it in a newly randomised set of feature and class name placeholders. For each item in the training set of TEXEN, the feature and class names were re-randomised ten times so that the augmented dataset contains 3421 records (train/validation/test split: 3280/47/94).

Validation and test sets do not undergo this augmentation process such that the direct comparison between models can occur.

Aside from the feature and class placeholders, the narratives will remain identical; the work does not attempt to rewrite the narratives in a new way. By re-randomising placeholders, it loosens the dependency on learning spurious correlations and encourages the model to learn the link between the features and feature values in the input and the features mentioned in the text.

4.4 Summarisation Task: Proposed Methodology

The purpose of the textual explanation generation task is to complement the existing numerical explanation pipeline, presenting a parallel pathway that converts the quantitative numerical explanations into accessible and easily understandable text. The goal is not to replace the numerical explanation, but to supplement it, communicating the same information that the graph or numerical summary conveys, but in a manner that is more intuitive for a non-expert audience.

This methods section will illustrate the proposed textual explanation pipeline, as shown in Figure 4.6, right side. This method first converts a dataset-specific input into one that is dataset agnostic, allowing for more generalisibility. Secondly, there is the trained language model and finally the post-processing that converts the anonymised text back into that which relates to the input features.

4.4.1 Problem Definition

Given a numerical explanation, the task is to produce a narrative that explains in text what the graph is showing. Formally, a numerical explanation consists of the following:

m class names

$$\mathbf{c} = [c_1, ..., c_m]\,, \tag{4.1}$$

their associated class probabilities

$$\mathbf{p} = [p_1, ..., p_m]\,, \tag{4.2}$$

n feature names

$$\mathbf{f} = [f_1, ..., f_n] \tag{4.3}$$

and their associated feature importance values

$$\mathbf{v} = [v_1, ..., v_n]\,. \tag{4.4}$$

The n^+ features with values $v_i \geq 0$ and the n^- features with values $v_j < 0$ such that $n^+ + n^- = n$ are formally defined as

$$\mathbf{f}^+ = \left[f_i^+, ..., f_{n^+}^+\right] \tag{4.5}$$

and

$$\mathbf{f}^- = \left[f_j^-, ..., f_{n^-}^-\right], \tag{4.6}$$

respectively, where $\mathbf{f}^+$ and $\mathbf{f}^-$ are subsets of $\mathbf{f}$, such that

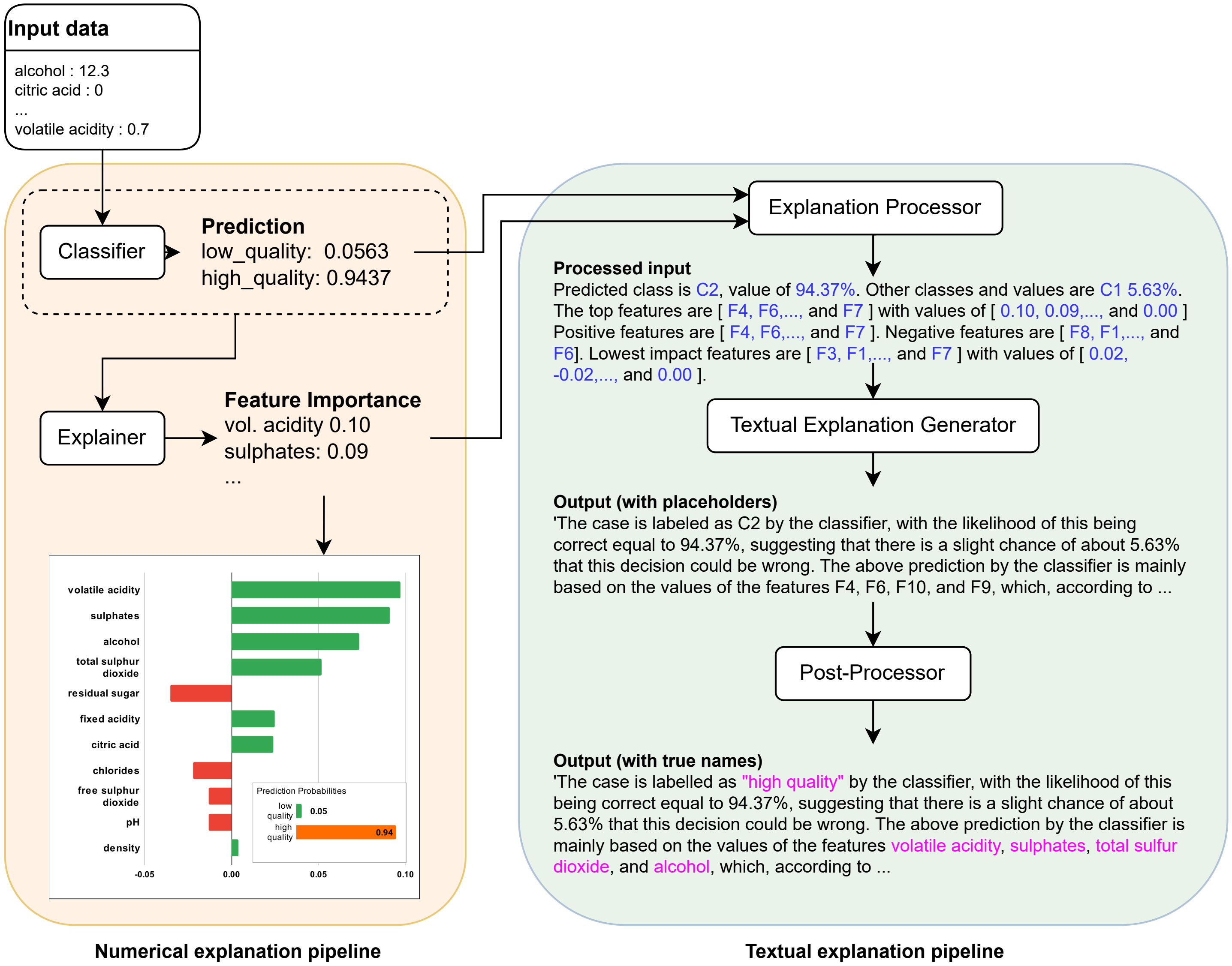
$$\mathbf{f} = \mathbf{f}^+ \cup \mathbf{f}^-. \tag{4.7}$$

4.4.2 Textual Explanation Pipeline

The proposed textual explanation pipeline has three components:

- An Explanation Processor that converts a numerical explanation into an input string with placeholder features and class names
- A language model trained for text-to-text generation
- A post-processor to replace the placeholders with actual feature and class names.

In this task, there are class names, probabilities, feature names and feature importance values for each input. Therefore, in order to use text-based language models, this structured data had to be formatted into an appropriate string template.



Processed input
Predicted class is C2, value of 94.37%. Other classes and values are C1 5.63%. The top features are [F4, F6,..., and F7] with values of [0.10, 0.09,..., and 0.00] Positive features are [F4, F6,..., and F7]. Negative features are [F8, F1,..., and F6]. Lowest impact features are [F3, F1,..., and F7] with values of [0.02, -0.02,..., and 0.00].

Output (with placeholders)
'The case is labeled as C2 by the classifier, with the likelihood of this being correct equal to 94.37%, suggesting that there is a slight chance of about 5.63% that this decision could be wrong. The above prediction by the classifier is mainly based on the values of the features F4, F6, F10, and F9, which, according to ...

Output (with true names)
'The case is labelled as "high quality" by the classifier, with the likelihood of this being correct equal to 94.37%, suggesting that there is a slight chance of about 5.63% that this decision could be wrong. The above prediction by the classifier is mainly based on the values of the features volatile acidity, sulphates, total sulfur dioxide, and alcohol, which, according to ...

Figure 4.6: A typical numerical explanation pipeline is on the left. The proposed, complimentary textual explanation pipeline is on the right. The predicted probabilities and feature importance values are processed into a template, values in blue. The Textual Explanation Generator is trained with placeholders, the Post-Processor replaces placeholders for actual names (in pink).

Explanation Processor

The input to the explanation processor is a numerical explanation, as defined above. At this stage **c**, **p**, **f** and **v** are reordered from highest absolute value to lowest to match the presentation of the output graphs. A set of class name placeholders $C1$, ..., Cm is shuffled and substituted in for each item in **c**. This approach is repeated for feature names, where each feature name in **f** is substituted for a placeholder in the shuffled set of $F1$, ..., Fn. Substitution is done so the model can transfer its learning from task to task; furthermore, tokenised inputs will not have to be truncated due to long feature names. This step is also crucial to prevent the model from learning from tasks it has seen before.

Following Moryossef et al. (2019); Chen et al. (2020a); Suadaa et al. (2021b), the final stage of the 'Explainer Processor' involves linearisation of the data into a flat string: **p**, **v** and the newly substituted **c** and **f** are formatted into the template below. A cap, top_n, set at $min(n, 10)$ or $min(n, 20)$ during training, is used to limit the number of top features passed into the model and positive and negative features are subsets of the capped top features, such that $top\_n^+ + top\_n^- = top\_n$; the lowest impact features are not affected. Note that the top features and values are formatted so that only the final value is preceded by 'and'.

> Predicted class is <c_1>, value of <p_1>. Other classes and values are <c_2> <p_2> & ... & <c_m> <p_m>. Top features are [<f_1>, ..., and <f_{top_n}>], with values [<v_1, ..., and <v_{top_n}>]. Positive features are [<f_i^+>, ..., and <$f^+_{top\_n^+}$>]. Negative features are [<f_j^->, ..., and $f^-_{top\_n^-}$>]. Lowest impact features are [<f_{n-4}>, ..., and <f_n>] with values [<v_{n-4}>, ..., and <v_n>].

Textual Explanation Generator

The tokenised inputs are passed into a pre-trained language model. This chapter experiments with both T5 and BART. These language models are trained in a sequence-to-sequence fashion, using the collected textual explanations (with placeholders substituted in) as reference texts. In training, this is the final stage. In testing, the output (with placeholders) is passed to the *Post-Processor*.

Post-Processor

The function of the post-processor is simply to reverse the placeholder substitution process. Using regular expressions, class and feature name placeholders are identified and mapped back to the original string values. This stage is not active during training when the model requires a consistent way of representing the data, but only during inference when it is helpful to report the true names.

4.4.3 Baseline

As a baseline for comparison, using the *base* models the input was translated into a fixed template style, similar to the model input but with values removed and set top_n as $min(n, 3)$:

> Predicted class is $<c_1>$, value of $<p_1>$. Other classes and values are $<c_2>$ $<p_2>$ & ... & $<c_m>$ $<p_m>$. Top features are [$<f_1>$, ..., and $<f_{top\_n}>$]. Positive features are [$<f_i^+>$, ..., and $<f_{top\_n^+}^+>$]. Negative features are [$<f_j^->$, ..., and $f_{top\_n^-}^->$]. Lowest impact features are [$<f_{n-4}>$, ..., and $<f_n>$].

4.5 Summarisation Task: Results

We fine-tune T5-base and BART-base models on the TEXEN and TEXEN-Augmented datasets. All experiments are run until validation performance has not increased for three epochs in a row. Once this limit has been reached, the best model is chosen, as decided by the lowest loss on the validation set. During inference, the neural generators generate textual explanations via beam search; examples of generated narrations are shown in the Error Analysis section (Fig. 4.7 and Fig. 4.8).

4.5.1 Automatic Evaluation

The quality of the output textual explanations is assessed using automatic metrics METEOR (Banerjee and Lavie, 2005), BLEU (Papineni et al., 2002), and BLEURT (Raffel et al., 2020). The BLEU and METEOR scores are employed to measure the surface-level similarity of the reference texts and the machine-generated text. On the other hand, the BLEURT score is a semantic equivalence-based metric that indicates how well the machine-produced text communicates the meaning of the reference text. A more detailed description of all evaluation metrics can be found in Section 2.3. We report the BLEU, BLEURT, and METEOR scores achieved on the test set in Table 4.3. Compared to the baselines, all models show a notably improved performance in all three reported metrics.

4.5.2 Error Analysis

We also conduct an error analysis on 30 records from the test set, generating narratives for each of our experiments and counting errors. Due to time constraints, we choose to focus on BART. We sifted through each sentence of each narration, classifying them as either:

- **Classification**: Talking about the predicted class probability
- **Top features**: Mentioning the most influential features
- **Named groups**: Referring to positive, negative, moderately influential or least influential features
- **Unnamed groups**: Typically of the form 'among these...' or 'all the remaining features...'

- **Summary**: General statements summarising the decision

If the sentence contained an error or did not make sense, then a one was tallied for that sentence, else zero. Table 4.4, for each of the sentence types, shows, for each model, how many times each sentence appeared and the proportion of sentences of that type that contained an error across the 30 analysed narrations.

Analysing the results, the model is more consistent at producing error-free sentences of certain types than others. ‘Classification’, and ‘Top features’ sentences are usually in a more consistent style in the collected narratives, which could be why the models were more successful at generating them. Using *top_n* of 10, rather than 20, tended to decrease the error rate, particularly in ‘Unnamed groups’, which the models found difficult. As shown in Fig. 4.7 and Fig. 4.8, the Textual Explanation Generators struggled with specific phrases that grouped or excluded previously mentioned features and made a claim about the said group.

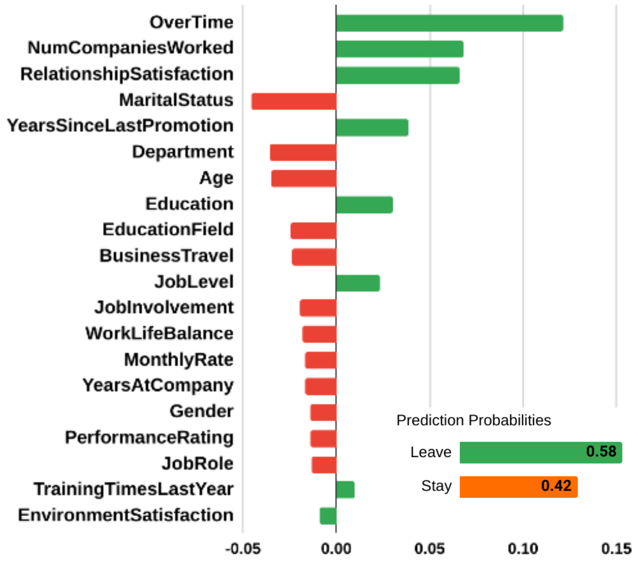
For all models except *base-10*, training on TEXEN-Augmented caused a lower error rate, demonstrating that providing more training data with re-randomised placeholders allows the model to learn the input-narrative relationship more effectively and make fewer false claims. Using BART-large also yielded a lower error rate, most notably in ‘Summary’ sentences where the generated narrative will tend to make broader statements without mentioning specific features, instead describing general patterns.

Table 4.3: Evaluation of textual explanation generation performance of the neural models. Avg. Rank refers to the mean in-column rank. Best in-column scores are in bold. (*base / large*) refers to base or large models, (*10 / 20*) refers to *top_n* and *Aug* refers to the use of TEXEN-Augmented, as opposed to TEXEN.

Experiment		BLEU	BLEURT	METEOR	Avg. Rank
BART	base-20	0.16	-0.25	0.36	6.7
	base-20-Aug	0.17	-0.23	0.36	**5.0**
	base-10	0.15	-0.19	0.34	8.0
	base-10-Aug	0.16	-0.23	0.36	6.3
	large-20	0.14	-0.28	**0.37**	9.3
	large-20-Aug	0.15	-0.27	0.35	10.7
	large-10	0.14	-0.25	0.34	12.7
	large-10-Aug	0.14	-0.26	0.35	10.3
	Baseline	0.08	-0.62	0.25	17.3
T5	base-20	0.16	-0.22	0.34	8.7
	base-20-Aug	0.17	-0.27	0.35	6.3
	base-10	0.17	-0.31	0.35	9.0
	base-10-Aug	0.17	-0.28	0.35	8.3
	large-20	0.17	**-0.17**	0.34	7.7
	large-20-Aug	**0.18**	-0.37	0.34	9.7
	large-10	**0.18**	-0.22	0.34	5.3
	large-10-Aug	0.17	-0.34	0.34	11.3
	Baseline	0.05	-0.62	0.25	17.7

Table 4.4: Error analysis of BART generated textual explanations per sentence type. Lowest error rate in bold. (*base* / *large*) refers to BART-base and BART-large, (*10* / *20*) refers to *top_n* and *Aug* refers to the use of TEXEN-Augmented, as opposed to TEXEN.

Experiment	Classification		Top		Unnamed Groups		Named Groups		Summary		Total	
	Count	Error Rate	Count	Error Rate	Count	Error Rate	Count	Error Rate	Count	Error Rate	Count	Error Rate
base-20	34	12%	29	14%	39	79%	17	53%	61	44%	180	42%
base-20-Aug	34	15%	30	**3%**	48	67%	39	41%	52	46%	203	38%
base-10	32	13%	30	13%	37	**49%**	25	52%	58	19%	182	27%
base-10-Aug	35	**6%**	30	10%	41	61%	29	62%	63	35%	198	35%
large-20	37	8%	32	22%	23	91%	42	43%	71	27%	205	33%
large-20-Aug	31	10%	29	7%	38	68%	21	**33%**	62	24%	181	29%
large-10	30	13%	34	15%	24	67%	31	42%	57	18%	176	27%
large-10-Aug	34	9%	29	14%	36	53%	30	40%	51	**14%**	180	**25%**

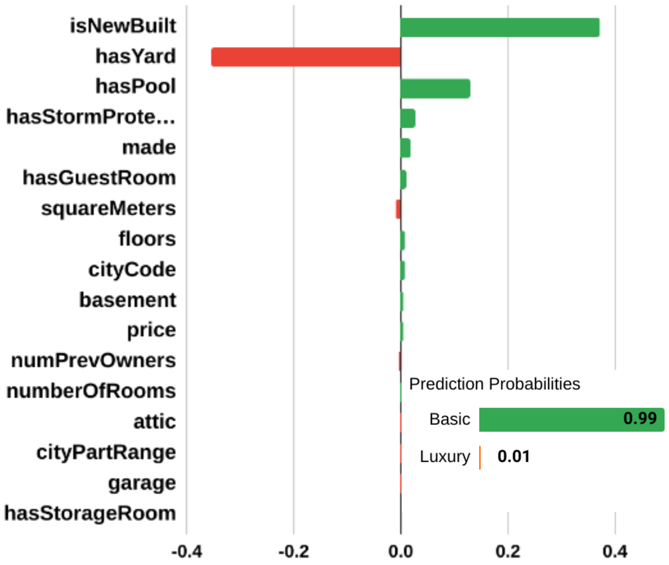


large-10-aug: The classifier is **57.83%** certain that **"Leave"** is the correct label for the case under consideration, implying that there is a **42.17%** chance that it could be **"Stay"** instead. The classification decision above is mainly based on the influence of features such as **OverTime**, **NumCompaniesWorked**, and **RelationshipSatisfaction**. Among these top influential features, only **MaritalStatus** has a negative impact, swinging the verdict in favour of any of the other labels. Other positive features with moderate influence include **Education**, **EducationField**, or **BusinessTravel**. However, considering the direction of effect of each input feature, it can be concluded that they are irrelevant to the prediction made here since their respective attributions are very close to zero. In simple terms, we can conclude that the model is not 100.0% confident in the assigned label given its level of certainty.

large-10: For the case under consideration, the model predicts class **"Leave"** with a probability of **57.83%**, implying that there is only a **42.17%** chance that **"Stay"** could be the correct label. The above prediction decision is mainly based on the values of **OverTime**, **NumCompaniesWorked**, and **RelationshipSatisfaction** which are shown to have very strong positive attributions, increasing the odds in favour of labelling the given case as **"Leave"**. On the contrary, all the remaining features such as **MaritalStatus**, **Department**, or **Age** negatively contribute to the decision made here. In terms of the direction of influence of each input feature, it can be concluded that the joint positive attribution outweighs the contributions of negative features hence the confidence level associated with the predicted label for this case.

Reference: 'The test case or example under consideration is labelled as **"Leave"** with a modest level of confidence because the probability that **"Stay"** is the correct label is approximately **42.17%**. The input features with the highest influence on the prediction above are **OverTime**, **NumCompaniesWorked**, **RelationshipSatisfaction**, **MaritalStatus**, **YearsSinceLastPromotion**, and **Department**, and those with moderate contributions are **Age**, **Education**, **EducationField**, **BusinessTravel**, **JobLevel**, **JobInvolvement**, and **WorkLifeBalance**. The influence of **MonthlyRate**, **YearsAtCompany**, **Gender**, **PerformanceRating**, **JobRole**, **TrainingTimesLastYear**, and **EnvironmentSatisfaction** can be described as moderately low. However, not all the features are considered by the classifier to arrive at the decision made for the given case. Irrelevant features include **YearsWithCurrManager**, **DailyRate**, **YearsInCurrentRole**, and **TotalWorkingYears**. The positive features driving the prediction in favour of the predicted label are **OverTime**, **NumCompaniesWorked**, **RelationshipSatisfaction**, **YearsSinceLastPromotion**, **Education**, **JobLevel**, and **EnvironmentSatisfaction**. Overall, the majority of the influential features have negative attributions that decrease the probability that **"Leave"** is the correct label, explaining the uncertainty associated with the prediction decision made by the classifier.'

Figure 4.7: Example of generated textual explanations and reference narration. The numerical explanation is shown here graphically on the left. The numerical explanation has feature and class names randomised and is fit into a template, as described in Section 4.4.2. Top right and centre show the output of two trained textual explainers, the difference being that large-10-aug has been trained on TEXEN-Augmented and large-10 trained on TEXEN. At the bottom, the reference text is shown. This is the narrative that was collected when the annotator was provided with the graph in this figure. Features, classes and values are in bold. Errors are in red.



large-10-aug: The classifier is **99.30%** certain that the correct label for this case is **"Basic"**, implying that there is only a **0.70%** chance that **"Luxury"** could be the true label. The classification decision above is mainly based on the influence of features such as **isNewBuilt**, **hasYard**, and **hasPool**. On the other hand, some of the least relevant features are shown to have negative attributions, shifting the verdict in the opposite direction. These negative features include **floors**, **cityCode**,and **basement**. However, considering the predicted likelihoods across the classes, it can be concluded that all the remaining features positively contribute to the prediction made here. As a matter of fact, the top positive features increasing the odds in favour of labelling the given case as **"Basic"**. In contrast, those with moderate contributions contradicting the model's decision regarding the appropriate label are **price**, or **cityPartRange**. To put it concisely, when you take into account the very high degree of certainty associated with the assigned label, one has to look at the attribution analysis to understand why the confidence level is quite high.

large-10: For the case under consideration, the model predicted **"Basic"** with **99.30%** certainty, implying that there is only a **0.70%** chance that **"Luxury"** could be the correct label. The prediction decision above is mainly based on the influence of the following features: **isNewBuilt**, **hasYard**, and **hasPool** which are shown to have very strong positive attributions, increasing the odds of labelling the given case as **"Basic"**. On the contrary, all the remaining features such as **hasStormProtector**, **made**, or **hasGuestRoom** negatively contribute to the prediction made here. All in all, it is valid to conclude that the classifier is very certain about the assigned label considering the degree of impact of each input feature.

Reference: Considering the values of the input features, the classifier generates the label **"Basic"** with **close to 100%** confidence, since the prediction probability of **"Luxury"** is only **0.70%**. The above classification judgement is mainly due to the influence of the features **isNewBuilt**, **hasPool**, and **hasYard** mainly because the classifier places more emphasis on their values than the remaining ones. Among these top features, **hasYard** is the one exhibiting negative influence, shifting the prediction decision towards the least probable class, **"Luxury"** and away from **"Basic"**. Conversely, **isNewBuilt** and **hasPool** are referred to as positive features since they increase the odds of the assigned **"Basic"** label instead of **"Luxury"**. Finally, unlike all the aforementioned, the values of **attic**, **cityPartRange**, **garage**, and **hasStorageRoom** have little impact on the classification output decision made here.

Figure 4.8: Another example of generated textual explanations and reference narration. Same format as in Fig. 4.7.

4.6 Q&A Task

4.6.1 Introduction

In the previous task, the goal was to generate a passage of text all at once. Here we look to simplify the task; we wanted to see how models would perform when we reduced the task to a question-answer problem. With this question-answer task, we use synthetically generated explanations for the training set and a variety of questions that address the information that annotators tended to speak about in the collected narrations.

4.6.2 Proposed Dataset

Here we investigate question-answering using synthetically generated numerical explanations by assigning random feature attributions and class values to class and feature placeholders. A training dataset of 27,000 records and a validation dataset of 3,000 records are generated in this manner. The question-answer pairs are created by randomly selecting a question from a pool of 8 templates for each numerical explanation. The test set consists of 469 records, using numerical explanations from the TEXEN train, validation, and test sets combined. For the test set, one question-answer pair is generated per numerical explanation.

Numerical explanations are synthetically generated in the following manner: Classes $C1$ and $C2$ have a random percentage probability (0.00%-100.00%) assigned to them, such that probabilities $p1 + p2 = 1$. top_n is set as a random number between 6-20, and then each of which is given a random feature placeholder and a random feature attribution between -0.50 and 0.50.

> Predicted class is <c_1>, value of <p_1>. Other classes and values are <c_2> <p_2> & ... & <c_m> <p_m>. Top features are [<f_1>, ..., and <f_{top_n}>], with values [<v_1, ..., and <v_{top_n}>]. Positive features are

[<f_i^+>, ..., and <$f_{top\_n^+}^+$>]. Negative features are [<f_j^->, ..., and $f_{top\_n^-}^-$>]. Lowest impact features are [<f_{n-4}>, ..., and <f_n>] with values [<v_{n-4}>, ..., and <v_n>]. Answer the following question: <Q>

The input string (above) is in the same format as in the textual explanation generation task but with an additional prompt and subsequent question, Q, which is selected at random from the eight question templates below. The questions are in bold.

Questions:

1. **What is the prediction for class *X*?** Class X is randomly chosen. The required answer is the predicted class probability for class X.

2. **What is the value of *X*?** X is a random feature name from the input. The answer is the value associated with feature X.

3. **Of the top *X* features, which are positive?** X is a random number between 2-5 inclusive. The task is to return the subset of the X most influential features that have a feature importance value greater than 0.

4. **Of the top *X* features, which are negative?** This follows the same pattern as above, but for feature importance values less than 0.

5. **Of these features [*ft_list*], which support the prediction?** *ft_list* is a list of 2-5 features, chosen at random from the input. The task is to return the subset of features from *ft_list* with a feature importance value greater than 0.

6. **Of these features [*ft_list*], which are against the prediction?** This follows the same pattern as above but for feature importance values less than 0.

7. **Which features have an absolute value greater than *X*?** *X* is a random float between 0.30-0.45 inclusive. The goal is to return a list of features with a value above *X*.

8. **Which are the *X* least important features?** *X* is a random number between 2-5 inclusive. The task is to return a list of the *X* features with the lowest feature importance scores.

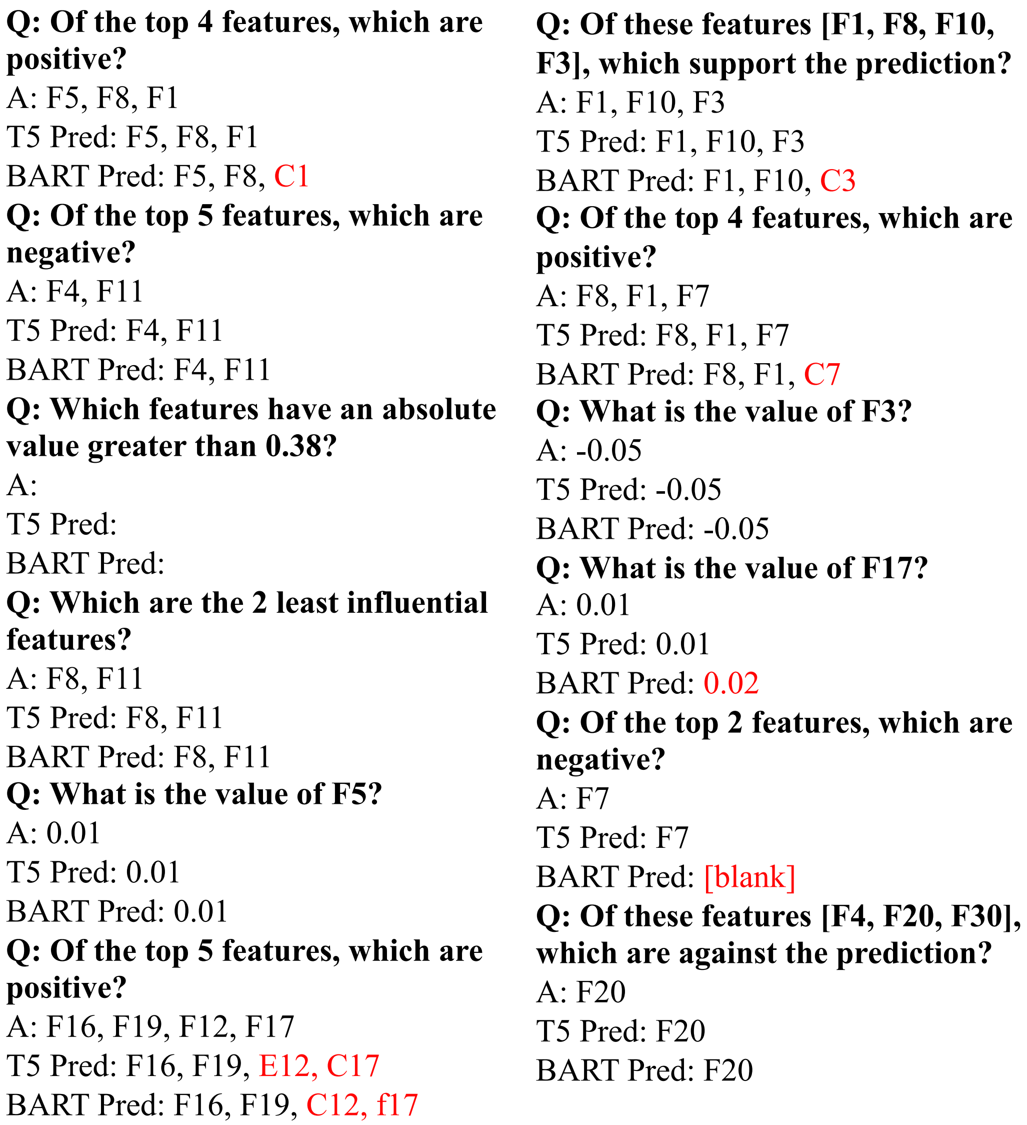
For questions 1 and 2, the answer is a single value, while the answers to questions 3-8 are lists of features separated by commas or blank if there is no correct answer.

4.6.3 Results

We train BART-base and T5-base models on the Question-Answer dataset and report the per-question accuracy in Table 4.5. Analysing the results, one can see that the models found some questions more straightforward; questions 1 and 2, which asked for a single class and feature value, scored the highest, perhaps because only a single figure was required instead of a list. For questions that need a list of numbers as an answer, if the generation matched the string exactly, then it was given a one, else zero. T5 scored especially highly, with an average accuracy of 91%. Examples are shown in Fig. 4.9

Table 4.5: Question Answer results

Question	**Accuracy**	
	BART-base	**T5-base**
Value of class X?	94%	100%
Value of feature X?	87%	99%
Of top X, which are positive?	59%	85%
Of top X, which are negative?	76%	97%
Of ft_list, which support?	62%	90%
Of ft_list, which are against?	82%	92%
Which features are >X?	73%	87%
X least important features?	39%	73%
Total	73%	91%



Q: Of the top 4 features, which are positive?
A: F5, F8, F1
T5 Pred: F5, F8, F1
BART Pred: F5, F8, C1

Q: Of the top 5 features, which are negative?
A: F4, F11
T5 Pred: F4, F11
BART Pred: F4, F11

Q: Which features have an absolute value greater than 0.38?
A:
T5 Pred:
BART Pred:

Q: Which are the 2 least influential features?
A: F8, F11
T5 Pred: F8, F11
BART Pred: F8, F11

Q: What is the value of F5?
A: 0.01
T5 Pred: 0.01
BART Pred: 0.01

Q: Of the top 5 features, which are positive?
A: F16, F19, F12, F17
T5 Pred: F16, F19, E12, C17
BART Pred: F16, F19, C12, f17

Q: Of these features [F1, F8, F10, F3], which support the prediction?
A: F1, F10, F3
T5 Pred: F1, F10, F3
BART Pred: F1, F10, C3

Q: Of the top 4 features, which are positive?
A: F8, F1, F7
T5 Pred: F8, F1, F7
BART Pred: F8, F1, C7

Q: What is the value of F3?
A: -0.05
T5 Pred: -0.05
BART Pred: -0.05

Q: What is the value of F17?
A: 0.01
T5 Pred: 0.01
BART Pred: 0.02

Q: Of the top 2 features, which are negative?
A: F7
T5 Pred: F7
BART Pred: [blank]

Q: Of these features [F4, F20, F30], which are against the prediction?
A: F20
T5 Pred: F20
BART Pred: F20

Figure 4.9: Example of questions, reference answers and predictions from both models. Errors are in red.

4.7 Discussion and Conlusion

As demonstrated by these two tasks, these models are able to provide extra clarity to assist in machine learning interpretability. While further comparisons could strengthen these conclusions, our principal aim was to introduce the task and methodology. We recognise that some may consider the dataset small; however, the difficulty of collecting quality narrations meant it was very costly and time-consuming to generate. As a result, this dataset represents the largest possible dataset we had the means to collect, and we are pleased to make it publicly available to benefit other researchers in the field.

The question-answer task was designed to cover the information held in numerical explanations; however, we acknowledge that the current set of questions may not cover all possible scenarios. Nevertheless, by using synthetic explanations, our dataset generation process allows for easy adaptation to encompass a new or expanded set of questions to suit specific needs.

In this work, we introduced a new NLG dataset of numerical-textual explanation pairs and trained T5 and BART to describe the output of feature importance-based explainers. When paired with the explainability graph, we aim to give users a better understanding of what the explanation means and, therefore, a better understanding of a given prediction decision. Automatic evaluation metrics show evidence of fluent explanations and error analysis yield reduced error rates when using TEXEN-Augmented. we also trained question-answer models for more structured answers and find T5-base gives us an overall accuracy of 91%. In the future, we plan to explore and utilise multi-modal modelling strategies, such as image captioning approaches, to directly use the explanation graphs without the linearisation steps.

Chapter 5

SHAP Explanations for Multimodal Text-Tabular Models

In this chapter, we address the research gap in explainability for multimodal machine learning, specifically for text-tabular data. We expose the errors that arise when pre-forming the data into text and applying the existing text masker and present a novel multimodal masking framework that extends SHAP to text-tabular datasets. In an extensive study, We examine the impact that combination strategies and language models have on SHAP explanations. Notably, the choice of combination method considerably influences the features identified as most important by the model. Furthermore, these findings reveal that methods converting all input to text tend to assign greater relative importance to text features over tabular features.

5.1 Introduction

As addressed in Chapter 2, despite the increasing popularity of multimodal tasks, there remains a gap in the research for producing explanations for these models. While there has been some work into text-image explainability (Parcalabescu and Frank, 2023; Lyu et al., 2022), text-tabular remains unexplored. A popular tool for explaining unimodal models - and explained in detail in Section 2.5.4 - is SHapley Additive exPlanations (SHAP) (Lundberg and Lee, 2017a), a game theory-based approach that relies on simulating coalitions of present and absent features. When multimodal data is fit to a single modality (for example, text, a method which we refer to *All-Text*) before the masking phase, it is possible to elicit an explanation. However, masking tabular features as if they were text can lead to erroneous token groupings and importance values assigned to non-feature tokens. To address this, we propose a novel multimodal masker, a complementary addition to the SHAP library. By fusing the previously separated text and tabular maskers and by deferring input formation until after the masking stage, we make it possible to generate SHAP explanations for any text-tabular model while also avoiding the pitfalls of the unimodal masker and the *All-Text* method. With this approach, text and tabular features are treated consistently, no matter how they are combined.

This framework, which is made publicly available, facilitates for the first time the generation of SHAP values for any text-tabular dataset and for any method of combining the two modalities. Moreover, we propose a series of experiments to compare the SHAP explanations of various combination methods on text-tabular datasets. By training four different text models, each on nine datasets and further with five combination methods, we intend not only to showcase the utility of the masker but also to gain insight as to which features drive performance, whether those features differ across experiments, and how reliance on each modality changes.

To summarise, the contributions of this chapter are the following:

- An exposure and analysis of the issues surrounding the current unimodal masker and an exploration of factors that may cause.
- A novel multimodal masking framework to allow the generation of SHAP values for any text-tabular dataset whilst ensuring that each modality is treated consistently.

- Extensive experimentation using the framework, assessing the effects changing text model and combination method have on nine different datasets.

5.2 Background: Limitations of Using Unimodal Masker for Multimodal Explanations

Generating SHAP explanations for multimodal inputs poses a challenge within the existing framework as it is inherently designed for unimodal data representations. Consequently, the generation of SHAP explanations is only possible in the *All-Text* scenario where one is training a text model and has already formed the data into a single modality (text). With any other approach, it is not possible to elicit a SHAP explanation. In addition to this limitation, we illustrate the problems that arise when using the unimodal masker for the *All-Text* method.

All-Text requires moulding an input into a string template, following a format such as *Column name*: *Column value*, delineated by '|'. However, when one forms this input prior to the masking call, the masker is unable to distinguish between tabular feature, text feature, or string template. Figure 5.1 (top) illustrates the process of using the unimodal masker and *All-Text*. As shown, each word piece of the tokenised input is now open to being selected as present or absent in a coalition. However, as the masker is treating the input as if it were a passage of text, tabular features are not sampled from a background dataset as is precedent. Instead, as demonstrated in the top right of the figure, tabular features can be split into multiple tokens and masked separately.

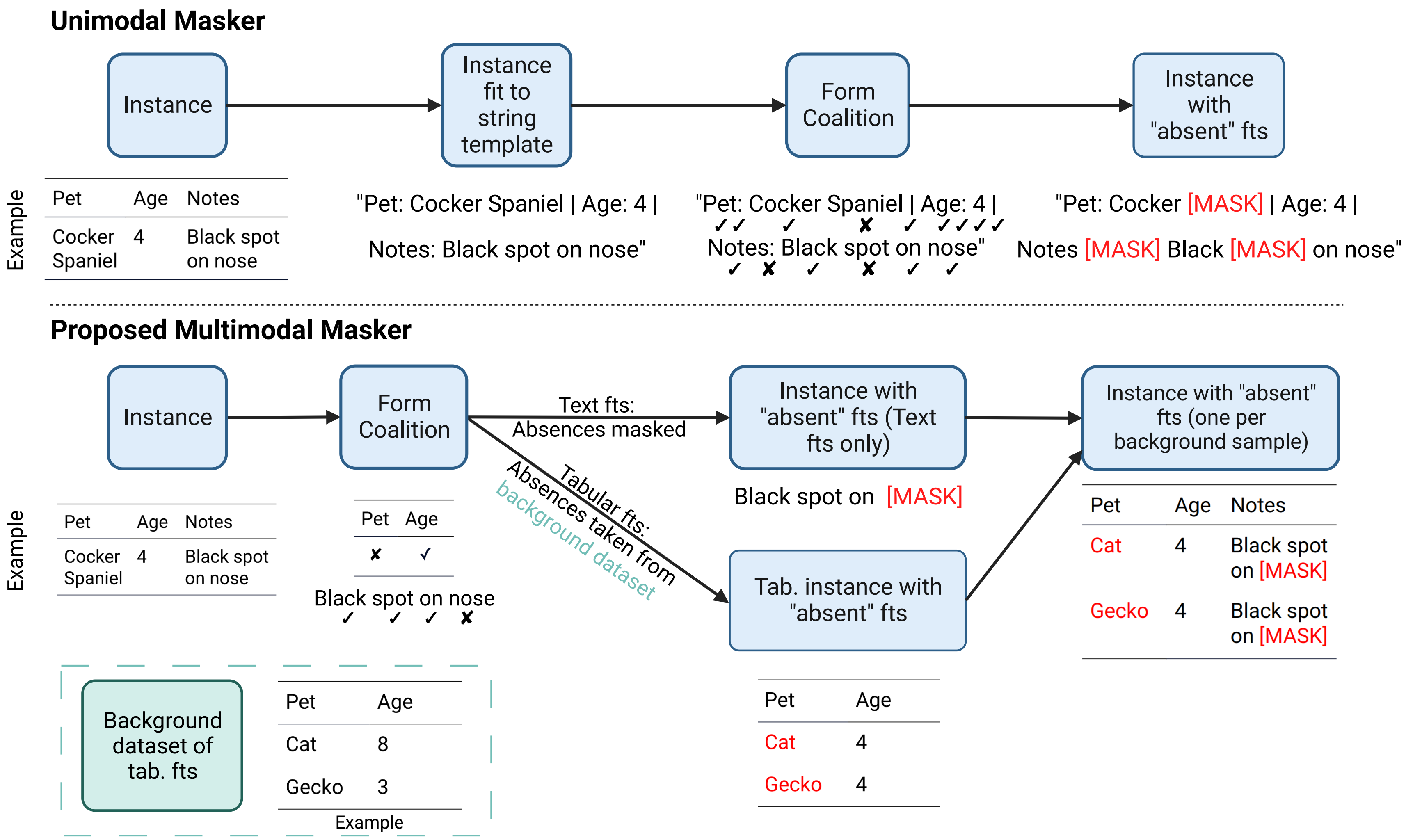
Furthermore, the string template, which is present and immutable for all instances and thereby necessarily has no impact between one instance and another, has no way of being recognised as such in the unimodal framework. Instead, as shown at the top of Figure 5.1 (top), template tokens will be treated as input features.

These issues are exacerbated further when the partition explainer is used, specifically during the formation of the partition hierarchy. The partition explainer saves compute by grouping tokens that form part of the same word or sentence together in order to reduce total number of computed coalitions. Details of the process are found in Section 2.5.5. Neighbouring tokens are grouped but do not take into account whether neighbours are members of the same or differing features, leading to incorrect grouping, separation of tokens and unrealistic and misleading explanations. In Figure 5.2, one can see each one of the numeric, tabular features being split up to form erroneous groups. We show a portion of the example below, with one group highlighted in red.

Year: 2010 | Runtime (Minutes): **98 | Rating: 6.**8 | Votes 157499 ...

When this group is absent, one is left with a misleading and unrealistic 8 minutes as a movie runtime.

Year: 2010 | Runtime (Minutes): 8 | Votes 157499 ...


Unimodal Masker
Instance
Instance fit to string template
Form Coalition
Instance with "absent" fts
Example
"Pet: Cocker Spaniel | Age: 4 | Notes: Black spot on nose"
"Pet: Cocker Spaniel | Age: 4 |
✓✓ ✓ ✘ ✓ ✓✓✓✓
Notes: Black spot on nose"
✓ ✘ ✓ ✘ ✓ ✓
"Pet: Cocker [MASK] | Age: 4 | Notes [MASK] Black [MASK] on nose"
Proposed Multimodal Masker
Instance
Form Coalition
Text fts: Absences masked
Tabular fts: Absences taken from background dataset
Instance with "absent" fts (Text fts only)
Instance with "absent" fts (one per background sample)
Tab. instance with "absent" fts
Example
Black spot on nose
✓ ✓ ✓ ✘
Black spot on [MASK]
Background dataset of tab. fts
Example

Pet	Age	Notes
Cocker Spaniel	4	Black spot on nose

Pet	Age
✘	✓

Pet	Age
Cat	8
Gecko	3

Pet	Age
Cat	4
Gecko	4

Pet	Age	Notes
Cat	4	Black spot on [MASK]
Gecko	4	Black spot on [MASK]

Figure 5.1: Comparison of masking processes with a multimodal text-tabular input, contrasting the current SHAP library's unimodal masker with the proposed multimodal masker. Using the unimodal masker in this scenario is only possible when the input is preformed into a string. **Top:** Each token in the string input can be either absent (swapped for [MASK] token) or present. Note that tabular features are divided into tokens (e.g., 'Cocker Spaniel'), and string template tokens, which should be omnipresent, can be absent(e.g., ':'). **Bottom:** Coalitions are formed, keeping tabular features as they are, while text features are split into tokens. Absent tabular features are sampled from a background dataset, while text tokens are replaced with [MASK] tokens. Once masked, features are reformed into an instance set, ready to be formatted for any multimodal model.

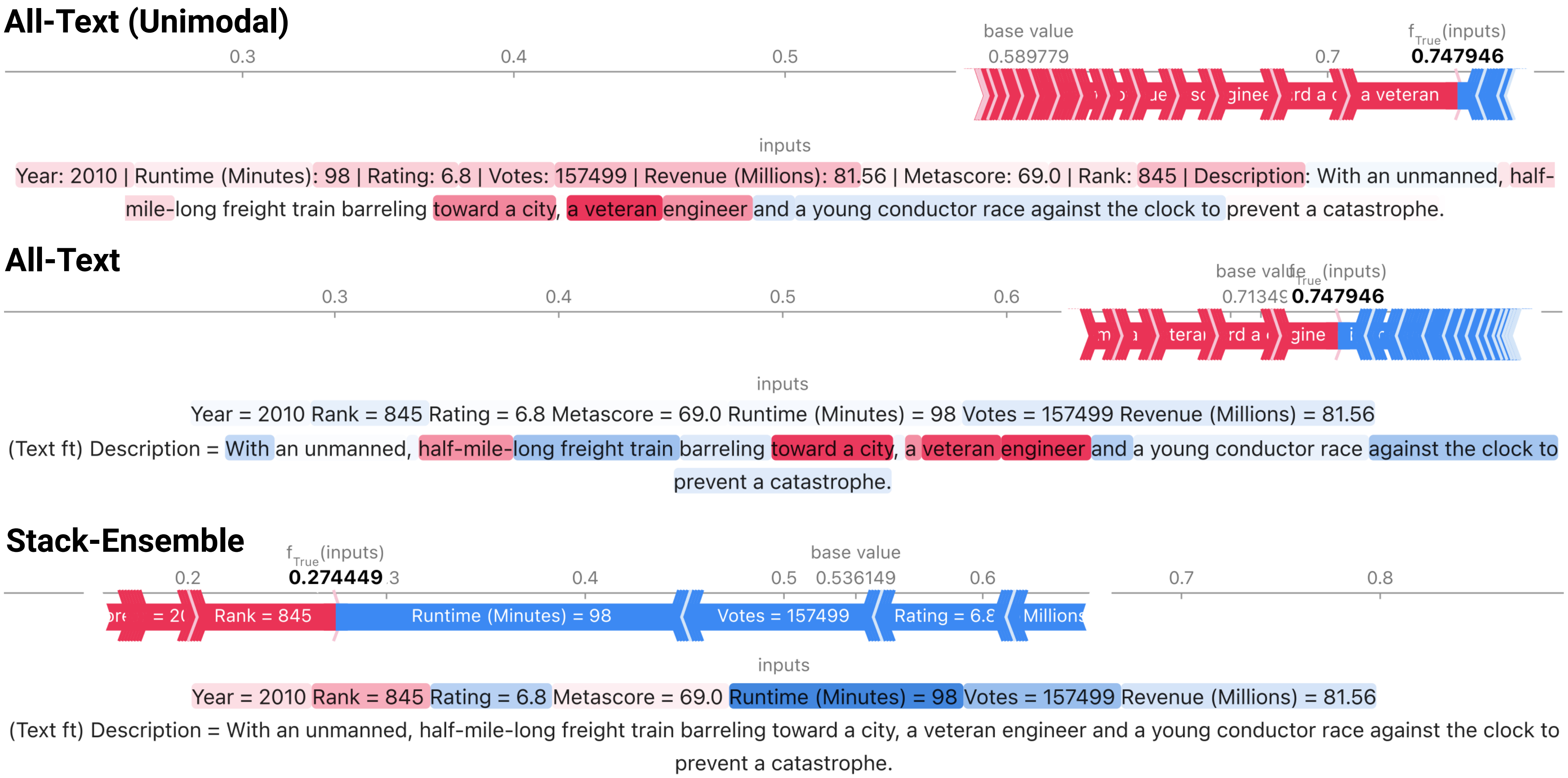


Figure 5.2: Three SHAP explanations of the same instance, all using text model= DistilBERT, dataset = *imdb*. Red indicates features that support the prediction, blue indicates those contributing in the opposite direction. Neighbouring work tokens with the same colour indicate membership of the same group for the given grouping threshold. **Top:** Using the original SHAP library's unimodal masker, combination method=*All-Text*. **Middle:** Using the proposed multimodal masker and combination method=*All-Text*. **Bottom:** Using the proposed multimodal masker and combination method=*Stack-Ensemble*.

5.3 Proposed Multimodal Masker

In this study, we introduce a novel multimodal masker that incorporates both text and tabular features without needing to first convert to a single modality in order to extend SHAP's capabilities for multimodal data. Our objective is to ensure that text and tabular features are treated consistently, no matter whether it is in an unimodal or multimodal scenario or which combination method is chosen. We condense these changes into a single masker class and a simple model wrapper such that it can be easily integrated into the existing SHAP framework. Finally, we also make adjustments to the SHAP plotter, as illustrated in Figure 5.2 (middle, bottom). For clarity, we plot tabular column names and values together and add a label to indicate which features are text. During experimentation, we generate SHAP values for each combination of three independent variables: combination method (CM), text model (TM), and dataset (DS). we analyse the resulting explanations; notably, we compare the explanations gathered by the proposed novel multimodal masker versus an unimodal one using the *All-Text* combination method.

Figure 5.1 (bottom) demonstrates the process of an explainer forming a coalition and then using the multimodal masker to form the model-compatible input. The key to the process is deferring the model input formation to after the masking call. In doing so, the proposed masker takes the unformatted features as input, separating out text and tabular features. In this process, text and tabular features are separated such that absent tokens text tokens are substituted for a mask token, and absent tabular features are sampled from a background dataset, as would be the case with unimodal data. This results in a masked instance set with present and absent features realised (one for each background dataset sample), as seen on the far right of the diagram. This set is then passed to the model prediction function, and the average prediction is used by the explainer to calculate the SHAP values. We wrap the model prediction with a simple function to convert the masked instance set to a model-compatible input. For example, one can divert tabular and text features to a tabular and text model, respectively, in a *Weighted-Ensemble*, or format each instance to a string template for the *All-Text* approach. In maintaining the separation, we make this process completely agnostic of the combination method. To ensure compatibility with the partition explainer, we add functionality to create a joint partition hierarchy for a multimodal input. Following precedent, tabular features use feature correlations, while neighbouring are grouped using the existing SHAP scoring method. The hybrid hierarchy is created by merging the two hierarchies at the highest level, thereby preventing the grouping of tabular features and word tokens.

5.4 Experimentation

5.4.1 Combination Methods

In the experimentation part of this study, we utilise five combination methods to merge text and tabular features, as illustrated in Figure 5.3: *Weighted-Ensemble* (using three different weights), *Stack-Ensemble*, and *All-Text*. These strategies are explained in this section.

Weighted-Ensemble: Tabular and text models are trained separately, and their predictions are combined with a weighted sum. We use w as the weight of the text model prediction and $(1 - w)$ as the weight of the tabular model prediction. In this study, the tabular model is a LightGBM (Ke et al., 2017), and we experiment with three values of w: 0.25, 0.50, and 0.75.

Stack-Ensemble: Similar to the *Weighted-Ensemble*, text and tabular models are trained independently on their respective features. However, in the *Stack-Ensemble*, a meta-model is introduced. This model is trained on the validation set using the predictions from both the tabular and text models, along with the original tabular inputs. The goal is to learn, based on the values of the tabular features, which predictions from either the tabular or text models should be given more emphasis. This approach allows for a more nuanced and adaptive combination of the predictions, potentially capturing interdependencies between the two modalities. LightGBMs are used as the tabular and meta models in our experiments. This strategy often performed best for Shi et al. (2021), although, in this study, we do not undertake k-fold cross-validation used to train the meta-model in order to reduce computation.

All-Text: In this method, all inputs, including tabular data, are treated as text, enabling us to leverage the capabilities of large language models. All features are fit to a string template and used to fine-tune a text model. In our experiments, we use a template in the form of *Column name*: *Column value*, delineated by '|'. An example is shown in Figure 5.2 (top).

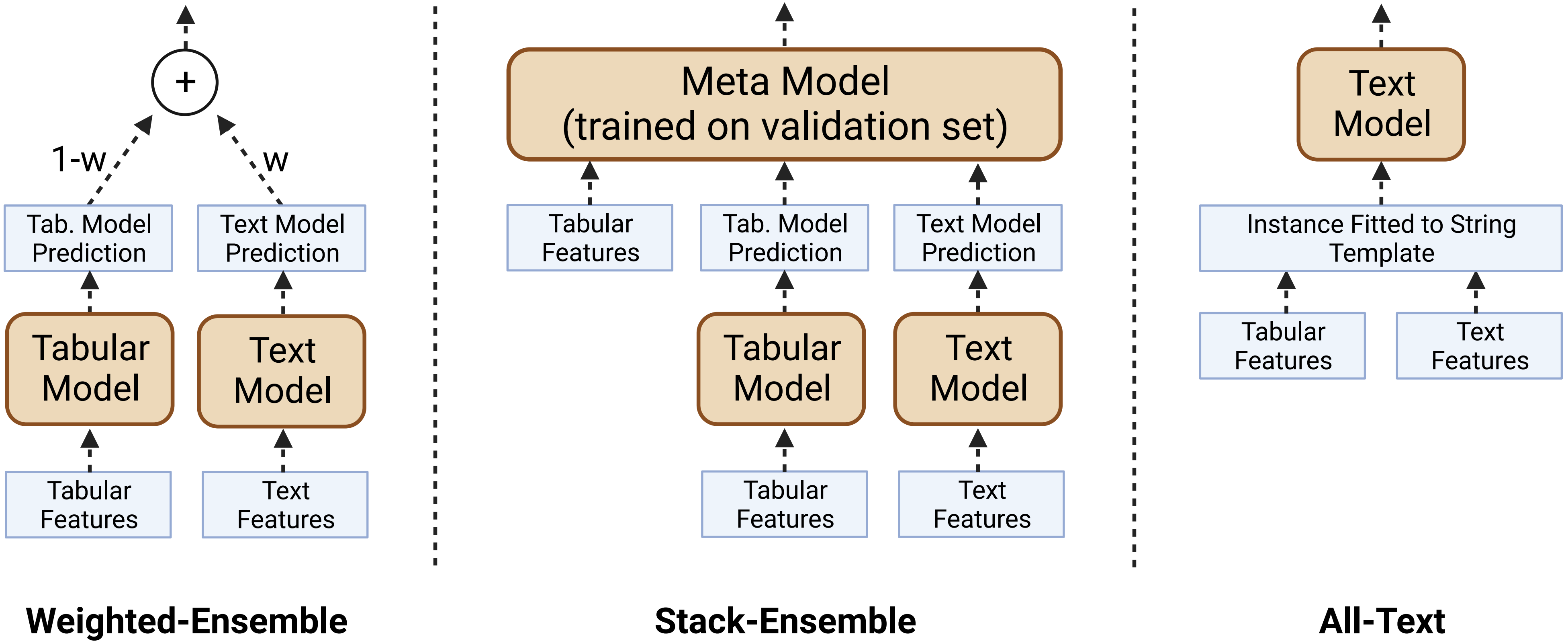


Figure 5.3: Combination methods used in this study. **Left:** A *Weighted-Ensemble*. **Middle:** A *Stack-Ensemble*. **Right:** *All-Text*.

5.4.2 Text Models

The pre-trained language models that we finetune are BERT (Devlin et al., 2018), DeBERTa (He et al., 2020), DistilBERT (Sanh et al., 2019), and DistilRoBERTa (Sanh et al., 2019). In order to isolate the differences to the independent variables in question, for each TM-DS, we only train two text models. One is trained for *All-Text*, which uses all features, and one, which only uses the tabular features, for the *Weighted-Ensembles* and *Stack-Ensemble*.

5.4.3 Datasets

The datasets used are from Shi et al. (2021), consisting of five multiclass and four binary classification tasks. We prepare the datasets in two different ways. The first, for the *All-Text* method, simply involves converting all values to strings. The second involves preparing the text and tabular features separately. Text features are converted to strings, numeric tabular features are left as they are, and categorical tabular features are ordinally encoded. For two of the datasets (*airbnb* and *imdb*), some of the features needed to be removed so as not to exceed the token limit of the text models. Preprocessing details and all final datasets are made available on the project's GitHub and Hugging Face repositories, respectively. The original test sets are used; the original train sets are divided into train and validation sets using an 85:15 split. After training the models, we find similar results to Shi et al. (2021) and report the results in Table 5.2. However, in some cases, our trained *Stack-Ensemble* models overfit and perform poorly on the test data. In order for our analysis of explanations to be valid, out of 216 experiments, we exclude 17 models that perform notably worse than others on the same dataset. For *channel*, we omit all *All-Text* and *All-Text (Unimodal)* results; for *salary* and *wine* we omit all *Stack-Ensemble* results and for *prod* we omit *Stack-Ensemble* for DistilBERT.

Table 5.1: Dataset information. This represents the dataset characteristics post-processing.

Dataset ID	Train	Test	#Tab. Fts	#Text Fts	Task	Metric	Prediction Target
prod	5,091	1,273	1	1	multiclass	accuracy	sentiment associated with product review
salary	15,841	3,961	1	5	multiclass	accuracy	salary range in data scientist job listings
airbnb	18,316	4,579	35	5	multiclass	accuracy	price label of Airbnb listing
channel	20,284	5,071	16	1	multiclass	accuracy	news category to which article belongs
wine	84,123	21,031	2	3	multiclass	accuracy	which variety of wine (type of grape)
imdb	800	200	7	1	binary	roc-auc	whether film is a drama
fake	12,725	3,182	2	3	binary	roc-auc	whether job postings are fake
kick	86,502	21,626	6	3	binary	roc-auc	whether Kickstarter project will achieve funding goal
jigsaw	100,000	25,000	29	1	binary	roc-auc	whether social media comments are toxic

Table 5.2: Performance of trained classifiers that are subsequently used to generate explanations. * indicates models that are excluded from explanation analysis due to poor performance relative to dataset peers. WE: Weighted-Ensemble

		airbnb	channel	fake	imdb	jigsaw	kick	prod	salary	wine
		(acc)	(acc)	(roc)	(roc)	(roc)	(roc)	(acc)	(acc)	(acc)
BERT	WE (w=.25)	.416	.544	.887	.851	.927	.748	.891	.431	.796
	WE (w=.50)	.421	.543	.928	.861	.941	.782	.881	.467	.826
	WE (w=.75)	.404	.502	.939	.857	.949	.776	.785	.479	.825
	Stack-Ensemble	.364	.466	.905	.822	.931	.755	.730	*.239	*.072
	All-Text	.387	*.254	.962	.828	.961	.781	.905	.481	.826
DeBERTa	WE (w=.25)	.418	.544	.871	.857	.927	.747	.891	.438	.781
	WE (w=.50)	.418	.540	.908	.872	.936	.779	.884	.468	.811
	WE (w=.75)	.400	.445	.921	.859	.944	.769	.811	.476	.810
	Stack-Ensemble	.351	.447	.905	.820	.928	.738	.874	*.277	*.078
	All-Text	.377	*.317	.959	.797	.955	.776	.888	.458	.817
DistilBERT	WE (w=.25)	.420	.545	.874	.853	.932	.741	.891	.394	.793
	WE (w=.50)	.419	.546	.919	.865	.946	.774	.879	.450	.822
	WE (w=.75)	.389	.481	.934	.852	.951	.768	.797	.456	.822
	Stack-Ensemble	.372	.449	.909	.811	.916	.749	*.665	*.171	*.022
	All-Text	.380	*.319	.961	.815	.962	.788	.901	.458	.819
DistilRoBERTa	WE (w=.25)	.419	.544	.865	.846	.941	.741	.891	.435	.798
	WE (w=.50)	.414	.547	.908	.853	.955	.784	.885	.468	.825
	WE (w=.75)	.387	.488	.927	.831	.961	.783	.796	.471	.824
	Stack-Ensemble	.380	.459	.919	.831	.903	.770	.885	*.329	*.037
	All-Text	.390	*.313	.958	.772	.964	.795	.904	.471	.821

5.4.4 Generating SHAP Values

For each dataset, we randomly select 100 instances from the test set to be explained. For experiments using the multimodal masker, a background dataset of size 100 (the default size for unimodal tabular SHAP) is randomly selected from the training set. Each TM-CM experiment on a dataset will explain the same 100 instances and use the same background dataset. We generate SHAP values for the selected instances for each TM-CM-DS combination using the multimodal masker. We also compute SHAP values for the same 100 instances for each TM-DS combination using the unimodal text masker and the *All-Text* combination method, we refer to these as *All-Text (Unimodal)*.

5.5 Results

We utilise a process similar to the SHAP package's summary plot function to account for the varying label counts and token quantities across instances. To be precise, for each instance, there are T tokens, each belonging to one of F features. Each token has associated SHAP values for L labels. First, we sum the SHAP values for each token, $t \in T$, belonging to a feature, $f \in F$. We then take the absolute value and sum across each label, $l \in L$. Condensing results in this manner yields a single SHAP value for each feature in each instance, indicating how important the feature was to the model. we refer to this as feature importance or ϕ.

$$\phi_f = \sum_{l \in L} |\sum_{t \in f} \text{SHAPValue}_{t,l}| \tag{5.1}$$

5.5.1 Template Analysis: Summary Plot Example

First, we look to understand how significant of an issue the problems are with *All-Text (Unimodal)*. Figure 5.4 shows a summary plot produced by the SHAP library for DS=*imdb*, TM=DisBERTa. The mean absolute SHAP value is charted for each of the labels (two, in this case). The average is across the whole set of 100 explained test set examples. Here, we have split up the input such that each input is made up of two parts: the feature template and the feature value. For example, here, the feature template is in red and the feature value is in blue for the Runtime (Minutes) feature:

'... | Runtime (Minutes): 98'

This summary plot is of the *imdb* dataset where the model predicts whether a film is a drama or not, so therefore, it is not surprising that the *Description* feature value is the most important and we also see the *Description* feature template as the second most important. However, the six next most important features are all feature templates. In fact, for all features bar *Description*, the feature template is

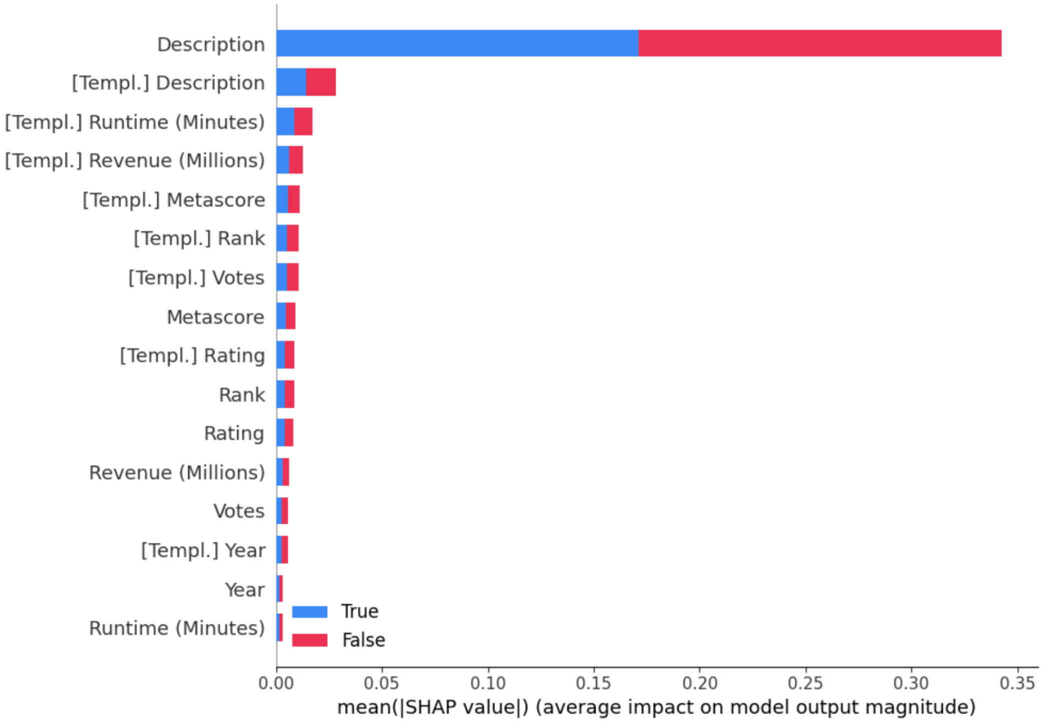


Figure 5.4: SHAP summary plot for *imdb* test set where TM=DisBERTa and CM=*All-Text (Unimodal)*. The mean absolute SHAP value is plotted on the x-axis for each of the features (y-axis). The features have been split up into two, with the template parts being labelled as [Templ.]. This is to demonstrate the erroneous importance being assigned to template values when using CM=*All-Text (Unimodal)*.

given a higher feature importance than the feature value despite being identical in every example.

Looking back to an individual SHAP plot in Figure 5.2, one can see the cause. As highlighted in Section 5.2, the template is being given an artificial level of importance with Unimodal SHAP asking the (invalid) question of ‘What happens if this feature template is absent?’ and simulating two numerical feature values conjoin. Figure 5.4 points towards this being the case for the entire dataset, not just an odd example.

5.5.2 Template Analysis: Summary Plot for All Experiments

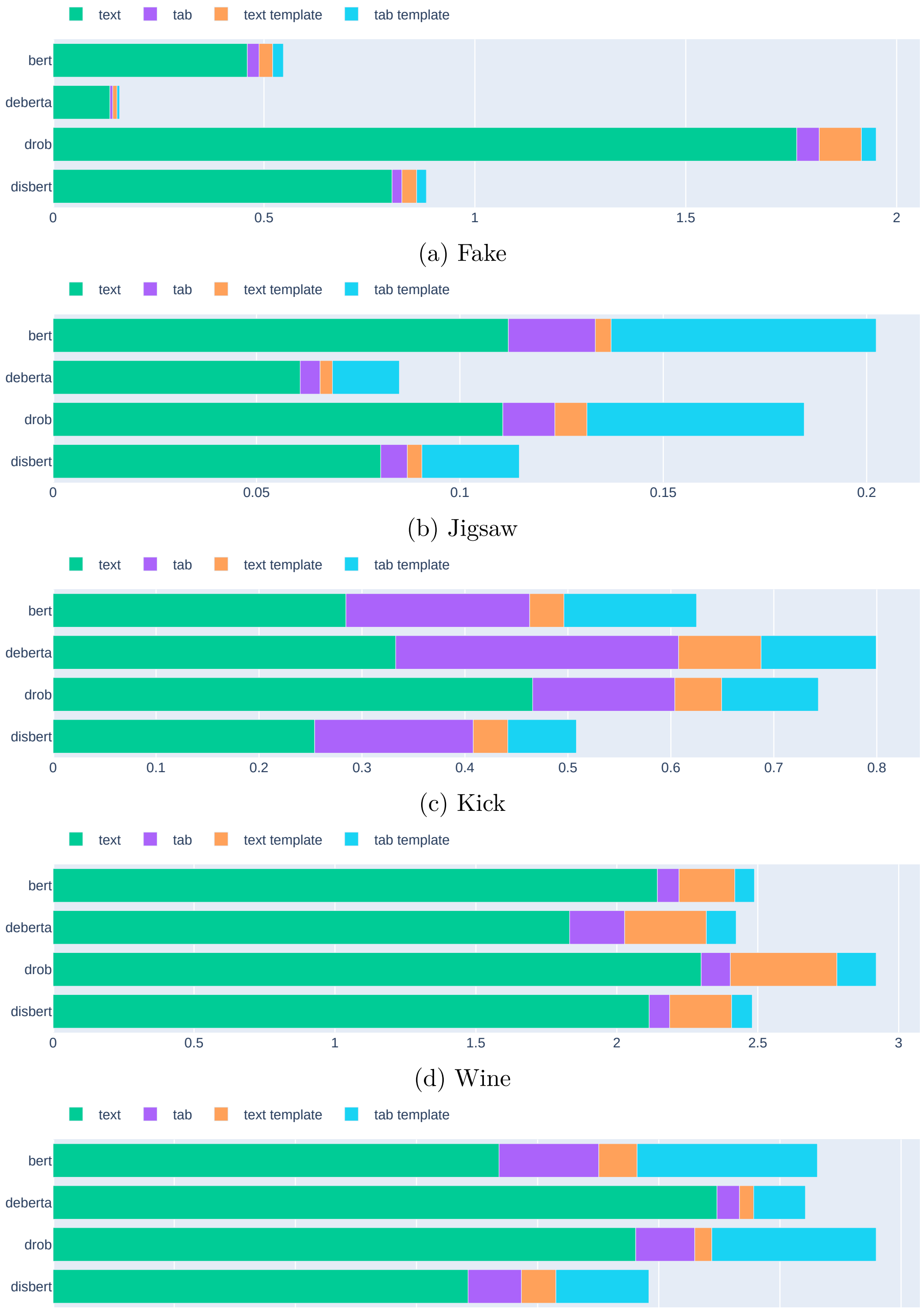
First, we wish to point out the relationship between the summary plot shown in Figure 5.4 and feature importance, ϕ. The summary plot shows the mean absolute feature importance for each feature and each label; in Figure 5.4, this is shown in blue for the *True* label and red for the False label. Note that for a binary problem, this is necessarily equal for both labels. ϕ is simply the mean absolute feature importance summed across labels; graphically, in this example, for each feature value and feature template, the red and the blue stacked bars become one.

In Figure 5.5, we stack ϕs for each template feature and template value and use the following colour scheme:

- Green: Text feature value
- Purple: Tabular feature value
- Orange: Text feature template
- Blue: Tabular feature template

Following the summary plot explored in Figure 5.4, this has now formed the bottom bars labelled ‘disbert’ in Figure 5.5(e). The only text feature for this dataset is *Description*; the previously blue and red mean absolute SHAP value for the feature value becomes a single green bar, and the *Description* template becomes the orange bar. All other features are tabular and they are summed to form the purple and blue bars for the feature values and feature templates, respectively.

Analysing the charts, one can see that this is not isolated to the *imdb* dataset. In fact, for most of the experiments, one can see that feature templates get assigned a substantial proportion of the total ϕ, tabular feature templates in particular. Indeed for TM=[*jigsaw, imdb, channel* and *airbnb*], the summed ϕ for the tabular feature templates exceeds that of the values. Furthermore, for *airbnb*, tabular feature templates carry higher importance than both text and tabular feature values.



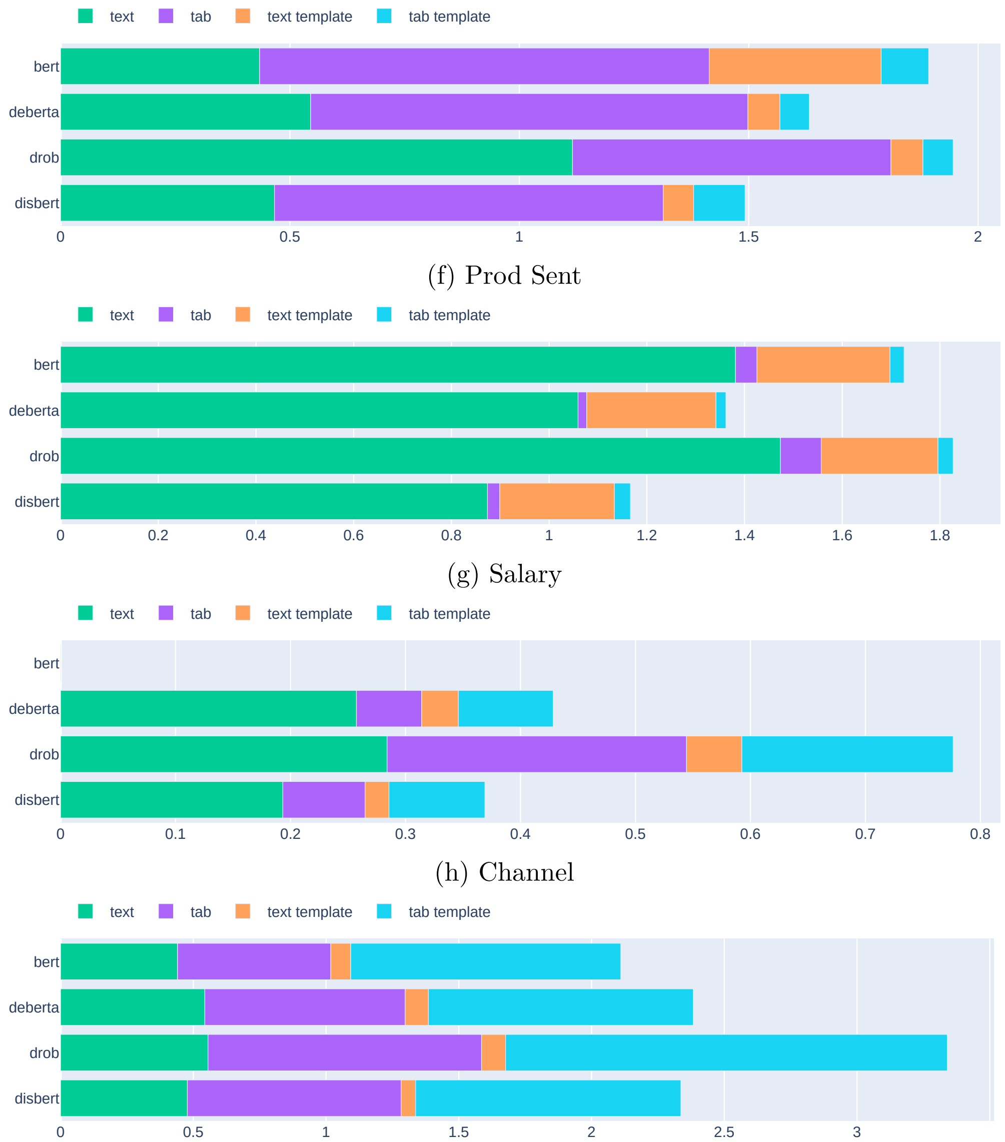
(a) Fake

(b) Jigsaw

(c) Kick

(d) Wine

(e) IMDb Genre



(f) Prod Sent

(g) Salary

(h) Channel

(i) Airbnb

Figure 5.5: Stacked mean feature importance charts for each of the four text models, for each of the nine datasets when CM=*All-Text (Unimodal)*. These are equivalent to the SHAP summary chart in Figure 5.4, but stacked horizontally. For each bar, each colour represents the level of importance assigned to text feature values (green) and templates (orange) and tabular features (purple) and templates (blue).

5.5.3 Template Analysis: Checking for Trends

Our goal was to explore any overarching reasons that would cause a template feature to be weighted more highly. We checked to see if there were any trends between

- Number of tokens in feature and template importance
- Feature value importance and template importance
- Feature value importance, split by text and tabular features, and absolute feature importance

To be clear, all of these experiments are done with *All-Text (Unimodal)* as this is the only combination method which yields feature importance values for templates. The other combination methods, that use the multimodal masker, do not assign importance to template tokens by design.

5.5.3.1 Is Template Feature Importance Affected by Length of Feature?

In Figure 5.6, each of the nine plots represents a point for each instance, for each feature for a particular dataset, coloured by text model. We use a log scale on both axes, considering that feature importances have a wide range. One can see no particular relationship between how long a particular feature is and how important the template is for that same feature.

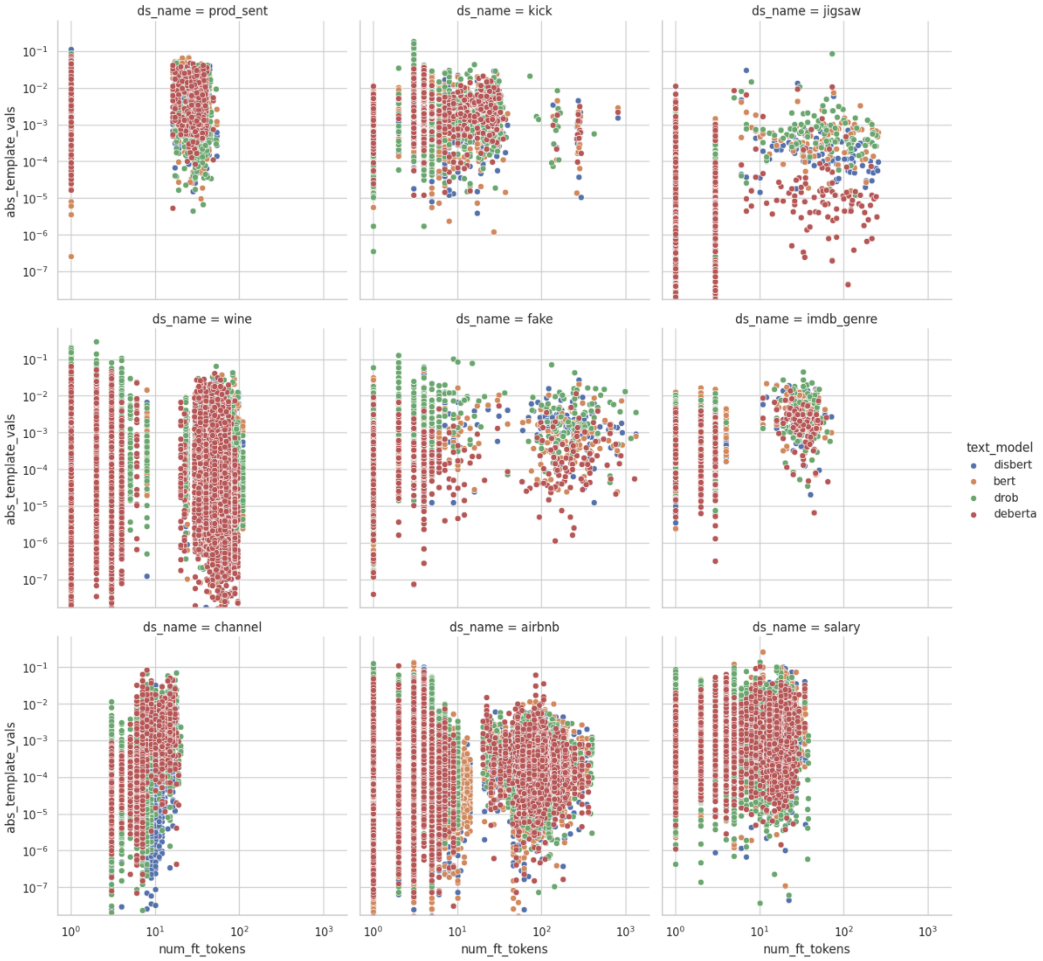


Figure 5.6: Scatter plot exploring if there is a relationship between the number of tokens in a feature (x-axis) with the absolute feature importance given to the template value of the measured feature (y-axis).

5.5.3.2 Does an Important Feature Value Lead to an Important Feature Template?

Similarly, in Figure 5.7, each of the nine plots represents a point for each instance, for each feature for a particular dataset, coloured by text model. Here, we hypothesised that a larger importance value for a feature's value would lead to a higher importance value for that feature's template. Some feature values were much longer than others, so therefore for each set of tokens in a feature value, we looked at the feature importance of each token and took the median value. This is plotted on the x-axis against the absolute value of the feature's template (y-axis).

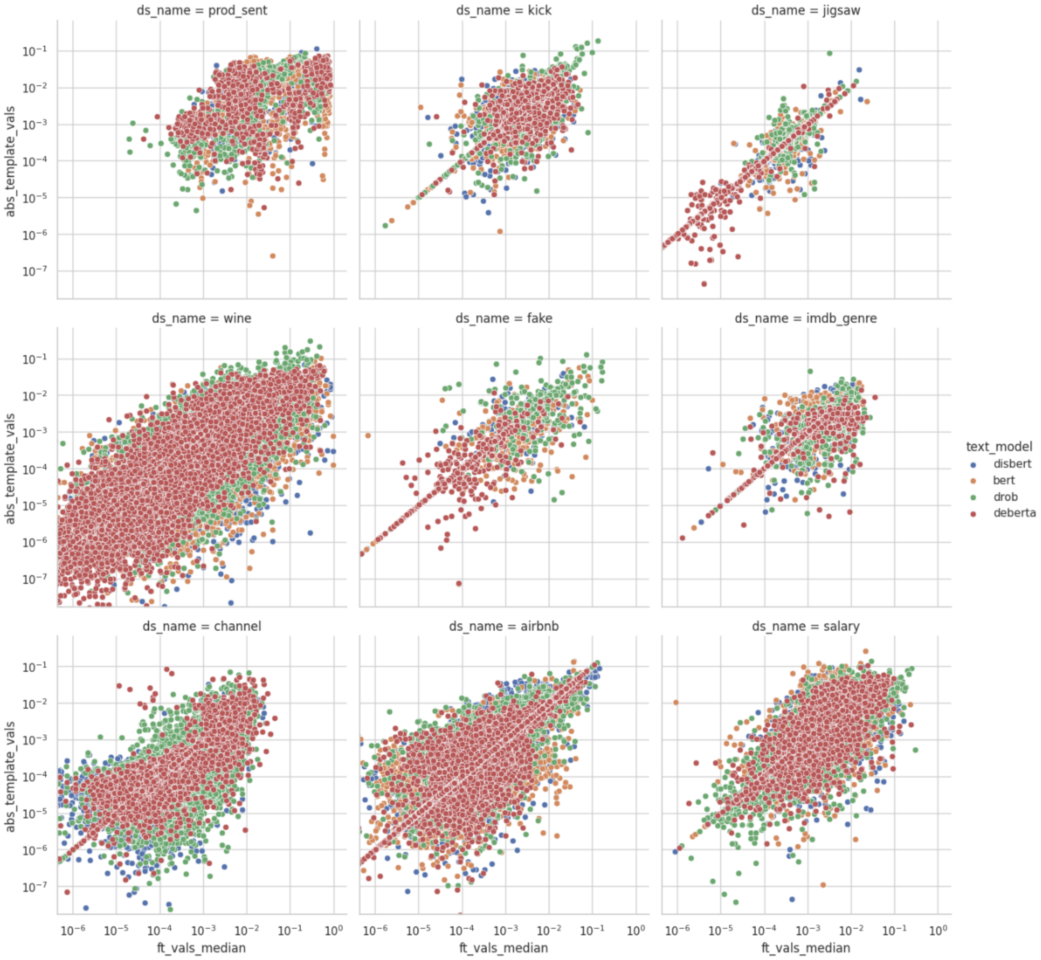


Figure 5.7: Scatter plot exploring if there is a relationship between median feature importance value number of tokens in a feature (x-axis) with the absolute feature importance given to the template value of the measured feature (y-axis).

Once more, a log scale is used for both axes. One can see a general trend between the two variables, although the severity of said relationship varies substantially between datasets with *wine* having a particularly loose fit.

5.5.3.3 Do Some Text Models or Combination Methods Assign More Importance to Template Tokens?

In Figure 5.5, we saw that when accumulated, tabular feature templates were often greater than the text feature equivalent. Therefore, we wanted to answer whether it was specifically because they were a tabular feature that they gained higher feature importance. In Figure 5.8, for each dataset, for each text model, we represent the distribution of absolute feature template importance for every feature of every instance with a box plot. Tabular features form red box plots and text features form blue box plots with green dots representing the mean. After Figure 5.5, we was expecting to see tabular template values score more highly than the text equivalent, however, this is not the case. In fact, although for most datasets the plots are very similar, for *jigsaw* and *channel* datasets the median text feature template is substantially higher than the tabular equivalent.

At first glance, this goes against what we saw in Figure 5.5; however, we theorise that this is down to the dissimilarity in the number of text and tabular features. Indeed, *jigsaw* and *channel* (together with *airbnb*) have the greatest disparity between the number of features for each modality. A higher median and interquartile range, as shown in the box plots, suggests a large number of these tabular features are unimportant and ignored, whereas *jigsaw* and *channel* only have a single text feature.

These results show that the more important a language model finds a feature, generally, the more important it will find the template value of the same feature. However, one cannot say that simply because a feature is one modality or another leads to a template value being assigned more or less importance.

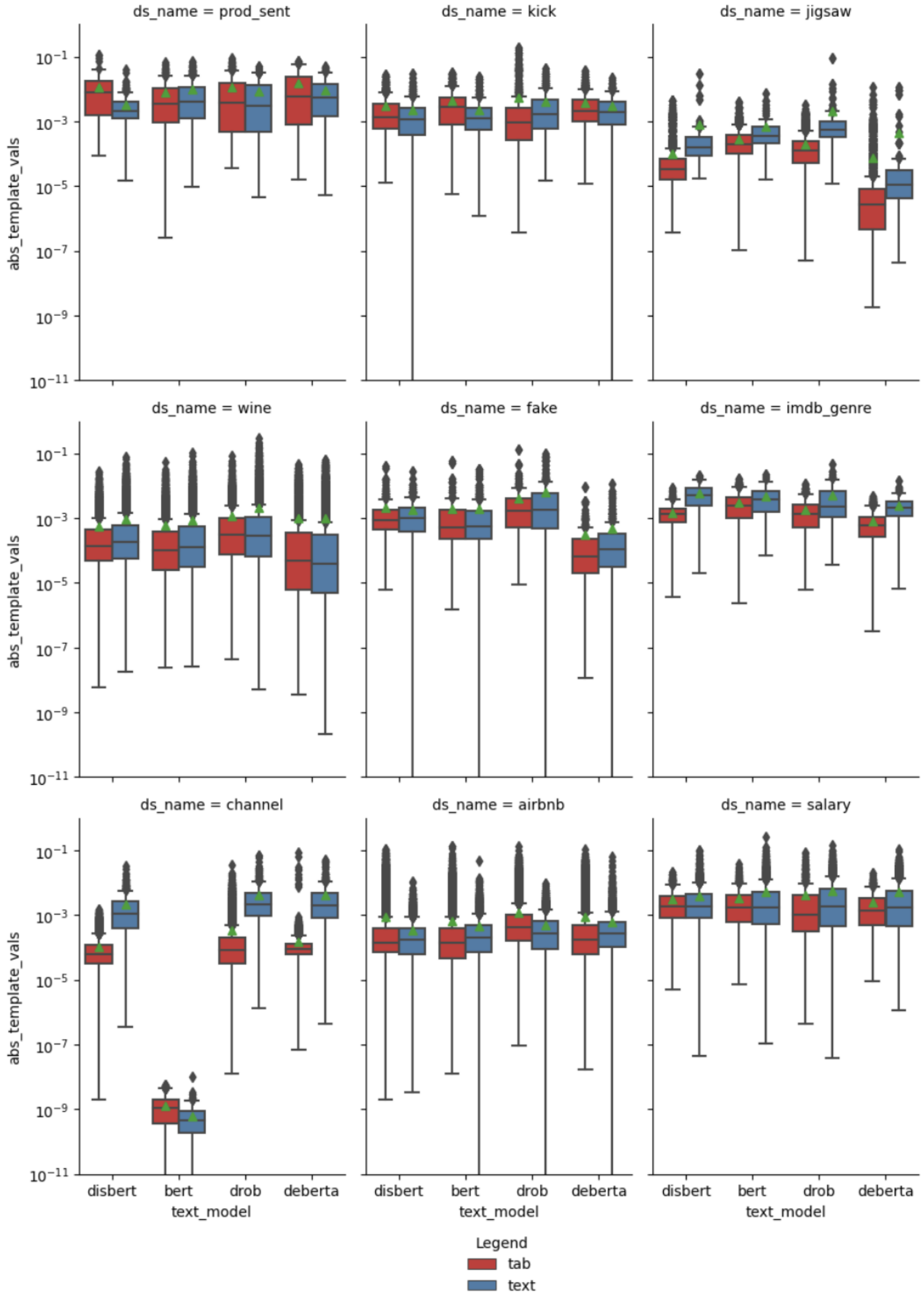


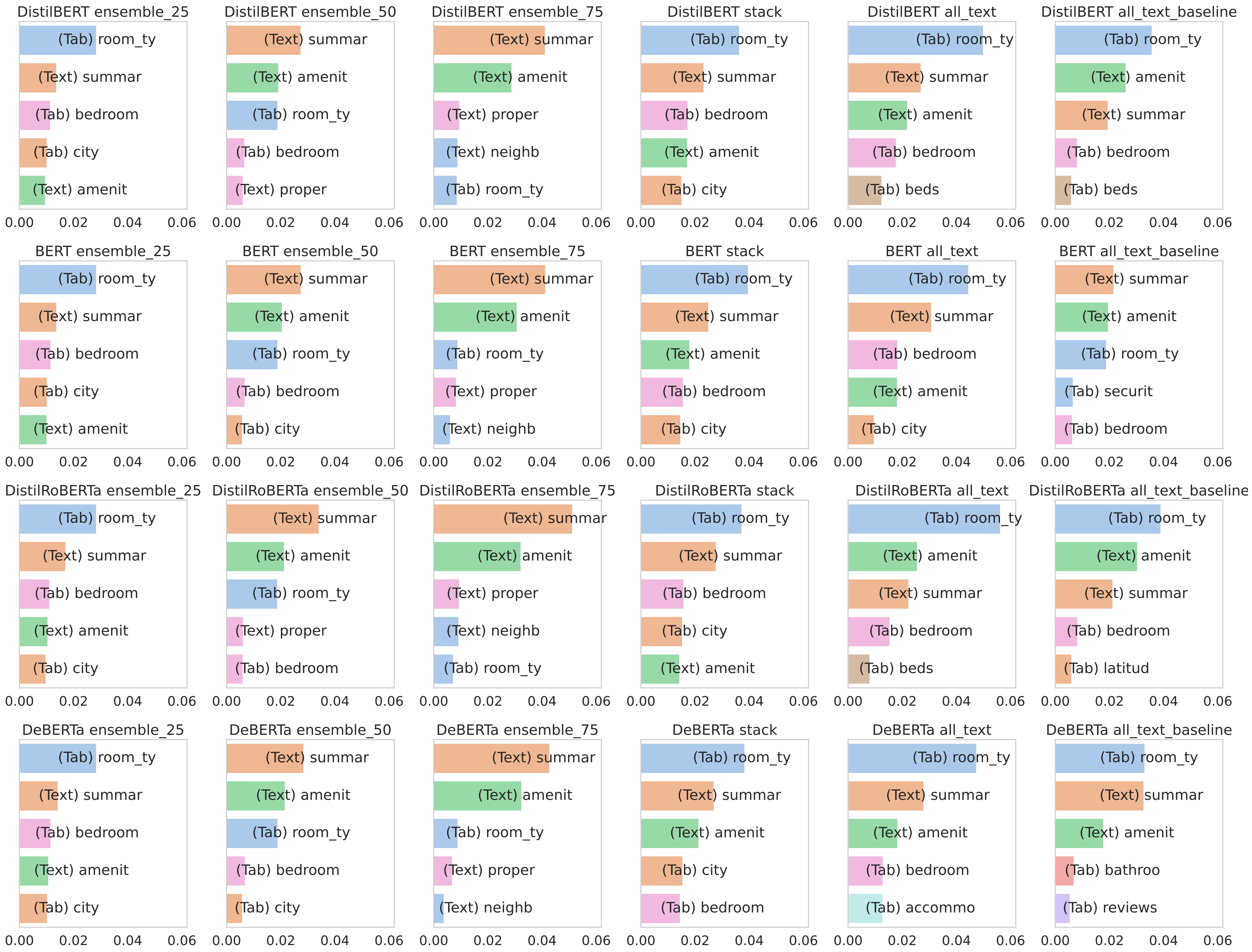
Figure 5.8: Absolute importance of a feature's template, against that feature's status as a text or tabular feature, further split by text model.

5.5.4 All Feature Analysis: Comparing Ranking of Features

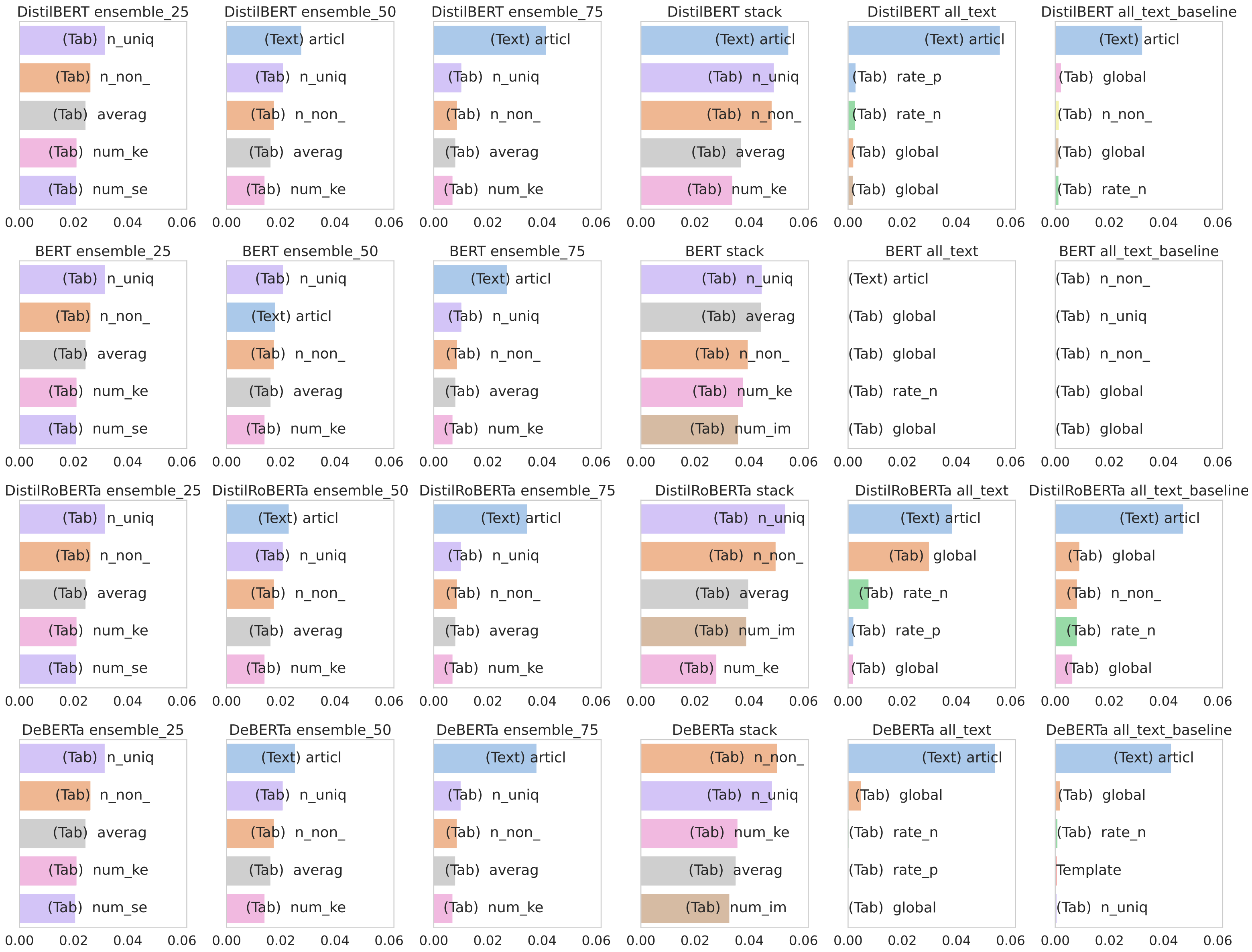
5.5.4.1 Top Five Features

We wish to answer the question: Are the same features always the most important? Specifically, for each dataset, we wanted to see whether the same inputs to the model are deemed as important when we change the underlying text model or combination method. For example, 'room type' is the most important feature for the *airbnb* dataset with text model as *DistilBERT* and combination method as *stack*, but is that still the case when we change to a different model? This will give an indication as to whether certain features in the data have an inherent importance, or alternatively does our choice of experiment determine which features are focused on? To make an assessment on this, first, we plotted bar charts of the top features by feature importance; here, the template is treated as a single feature. This is shown in Figures 5.9.

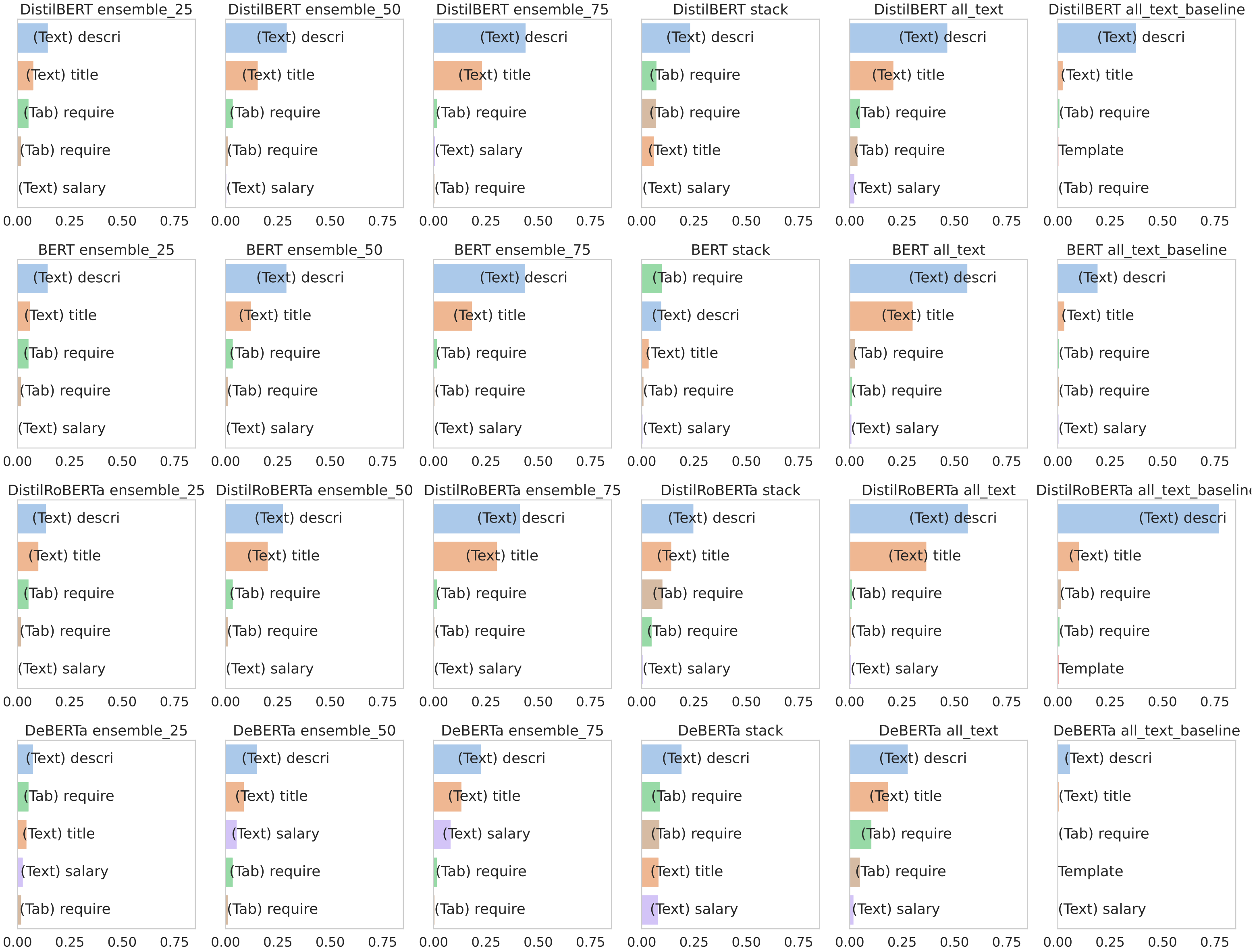
These sets of charts offer three conclusions: first, there are certainly differences in how features are ranked; second, the scale of the ϕ values vary across experiments; and third, within experiments, ϕ values do not appear to be normally distributed across features, frequently with a single feature much more important than the others, and/or a long tail of unimportant features, although this is not shown in the charts which only graph the top five for each experiment.


DistilBERT ensemble_25
(Tab) room_ty
(Text) summar
(Tab) bedroom
(Tab) city
(Text) amenit
DistilBERT ensemble_50
(Text) summar
(Text) amenit
(Tab) room_ty
(Tab) bedroom
(Text) proper
DistilBERT ensemble_75
(Text) summar
(Text) amenit
(Text) proper
(Text) neighb
(Tab) room_ty
DistilBERT stack
(Tab) room_ty
(Text) summar
(Tab) bedroom
(Text) amenit
(Tab) city
DistilBERT all_text
(Tab) room_ty
(Text) summar
(Text) amenit
(Tab) bedroom
(Tab) beds
DistilBERT all_text_baseline
(Tab) room_ty
(Text) amenit
(Text) summar
(Tab) bedroom
(Tab) beds
BERT ensemble_25
(Tab) room_ty
(Text) summar
(Tab) bedroom
(Tab) city
(Text) amenit
BERT ensemble_50
(Text) summar
(Text) amenit
(Tab) room_ty
(Tab) bedroom
(Tab) city
BERT ensemble_75
(Text) summar
(Text) amenit
(Tab) room_ty
(Text) proper
(Text) neighb
BERT stack
(Tab) room_ty
(Text) summar
(Text) amenit
(Tab) bedroom
(Tab) city
BERT all_text
(Tab) room_ty
(Text) summar
(Tab) bedroom
(Text) amenit
(Tab) city
BERT all_text_baseline
(Text) summar
(Text) amenit
(Tab) room_ty
(Tab) securit
(Tab) bedroom
DistilRoBERTa ensemble_25
(Tab) room_ty
(Text) summar
(Tab) bedroom
(Text) amenit
(Tab) city
DistilRoBERTa ensemble_50
(Text) summar
(Text) amenit
(Tab) room_ty
(Text) proper
(Tab) bedroom
DistilRoBERTa ensemble_75
(Text) summar
(Text) amenit
(Text) proper
(Text) neighb
(Tab) room_ty
DistilRoBERTa stack
(Tab) room_ty
(Text) summar
(Tab) bedroom
(Tab) city
(Text) amenit
DistilRoBERTa all_text
(Tab) room_ty
(Text) amenit
(Text) summar
(Tab) bedroom
(Tab) beds
DistilRoBERTa all_text_baseline
(Tab) room_ty
(Text) amenit
(Text) summar
(Tab) bedroom
(Tab) latitud
DeBERTa ensemble_25
(Tab) room_ty
(Text) summar
(Tab) bedroom
(Text) amenit
(Tab) city
DeBERTa ensemble_50
(Text) summar
(Text) amenit
(Tab) room_ty
(Tab) bedroom
(Tab) city
DeBERTa ensemble_75
(Text) summar
(Text) amenit
(Tab) room_ty
(Text) proper
(Text) neighb
DeBERTa stack
(Tab) room_ty
(Text) summar
(Text) amenit
(Tab) city
(Tab) bedroom
DeBERTa all_text
(Tab) room_ty
(Text) summar
(Text) amenit
(Tab) bedroom
(Tab) accommo
DeBERTa all_text_baseline
(Tab) room_ty
(Text) summar
(Text) amenit
(Tab) bathroo
(Tab) reviews
0.00
0.02
0.04
0.06

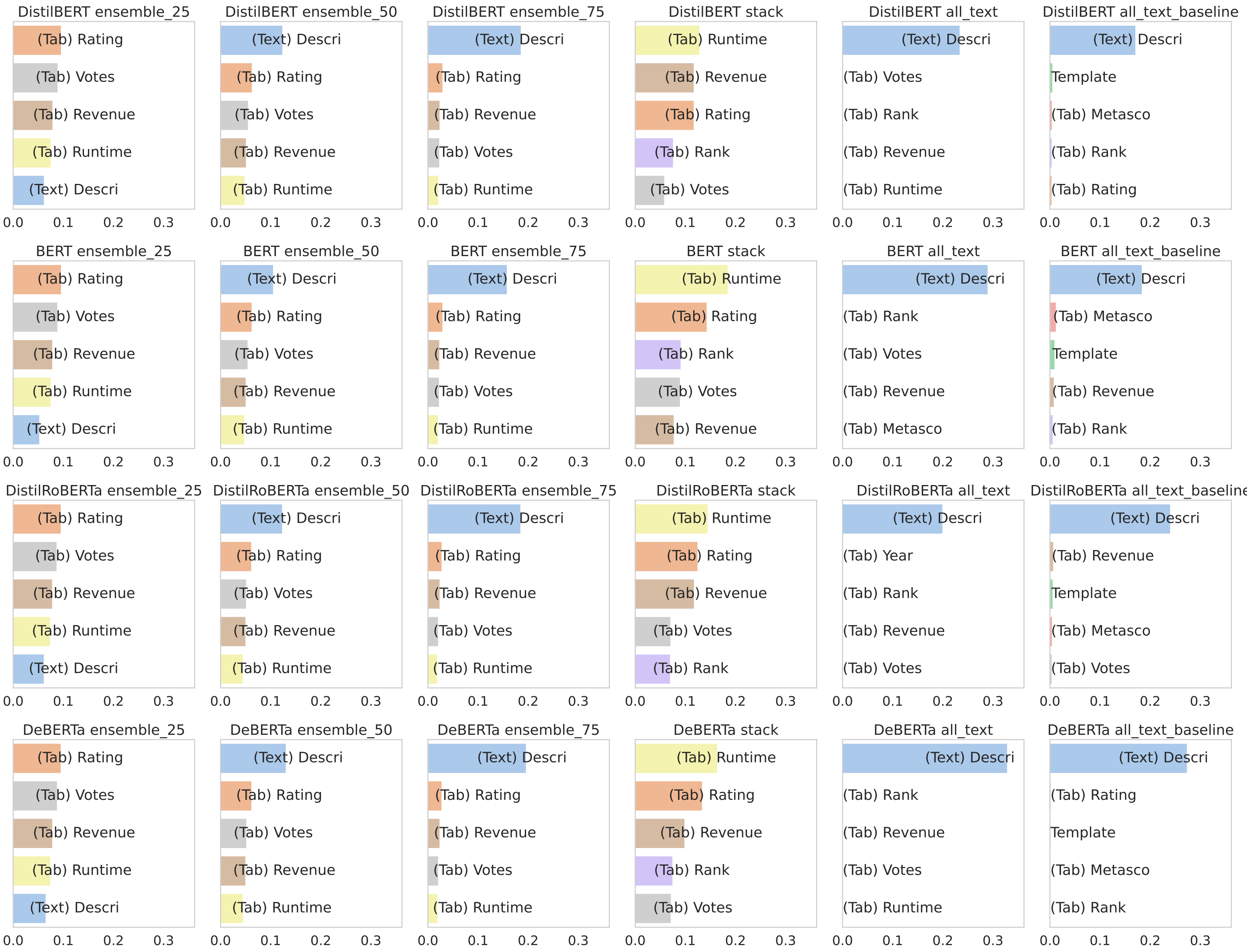
(a) Airbnb


DistilBERT ensemble_25
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
(Tab) num_se
DistilBERT ensemble_50
(Text) articl
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
DistilBERT ensemble_75
(Text) articl
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
DistilBERT stack
(Text) articl
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
DistilBERT all_text
(Text) articl
(Tab) rate_p
(Tab) rate_n
(Tab) global
(Tab) global
DistilBERT all_text_baseline
(Text) articl
(Tab) global
(Tab) n_non_
(Tab) global
(Tab) rate_n
BERT ensemble_25
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
(Tab) num_se
BERT ensemble_50
(Tab) n_uniq
(Text) articl
(Tab) n_non_
(Tab) averag
(Tab) num_ke
BERT ensemble_75
(Text) articl
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
BERT stack
(Tab) n_uniq
(Tab) averag
(Tab) n_non_
(Tab) num_ke
(Tab) num_im
BERT all_text
(Text) articl
(Tab) global
(Tab) global
(Tab) rate_n
(Tab) global
BERT all_text_baseline
(Tab) n_non_
(Tab) n_uniq
(Tab) n_non_
(Tab) global
(Tab) global
DistilRoBERTa ensemble_25
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
(Tab) num_se
DistilRoBERTa ensemble_50
(Text) articl
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
DistilRoBERTa ensemble_75
(Text) articl
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
DistilRoBERTa stack
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_im
(Tab) num_ke
DistilRoBERTa all_text
(Text) articl
(Tab) global
(Tab) rate_n
(Tab) rate_p
(Tab) global
DistilRoBERTa all_text_baseline
(Text) articl
(Tab) global
(Tab) n_non_
(Tab) rate_n
(Tab) global
DeBERTa ensemble_25
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
(Tab) num_se
DeBERTa ensemble_50
(Text) articl
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
DeBERTa ensemble_75
(Text) articl
(Tab) n_uniq
(Tab) n_non_
(Tab) averag
(Tab) num_ke
DeBERTa stack
(Tab) n_non_
(Tab) n_uniq
(Tab) num_ke
(Tab) averag
(Tab) num_im
DeBERTa all_text
(Text) articl
(Tab) global
(Tab) rate_n
(Tab) rate_p
(Tab) global
DeBERTa all_text_baseline
(Text) articl
(Tab) global
(Tab) rate_n
Template
(Tab) n_uniq
0.00
0.02
0.04
0.06

(b) Channel

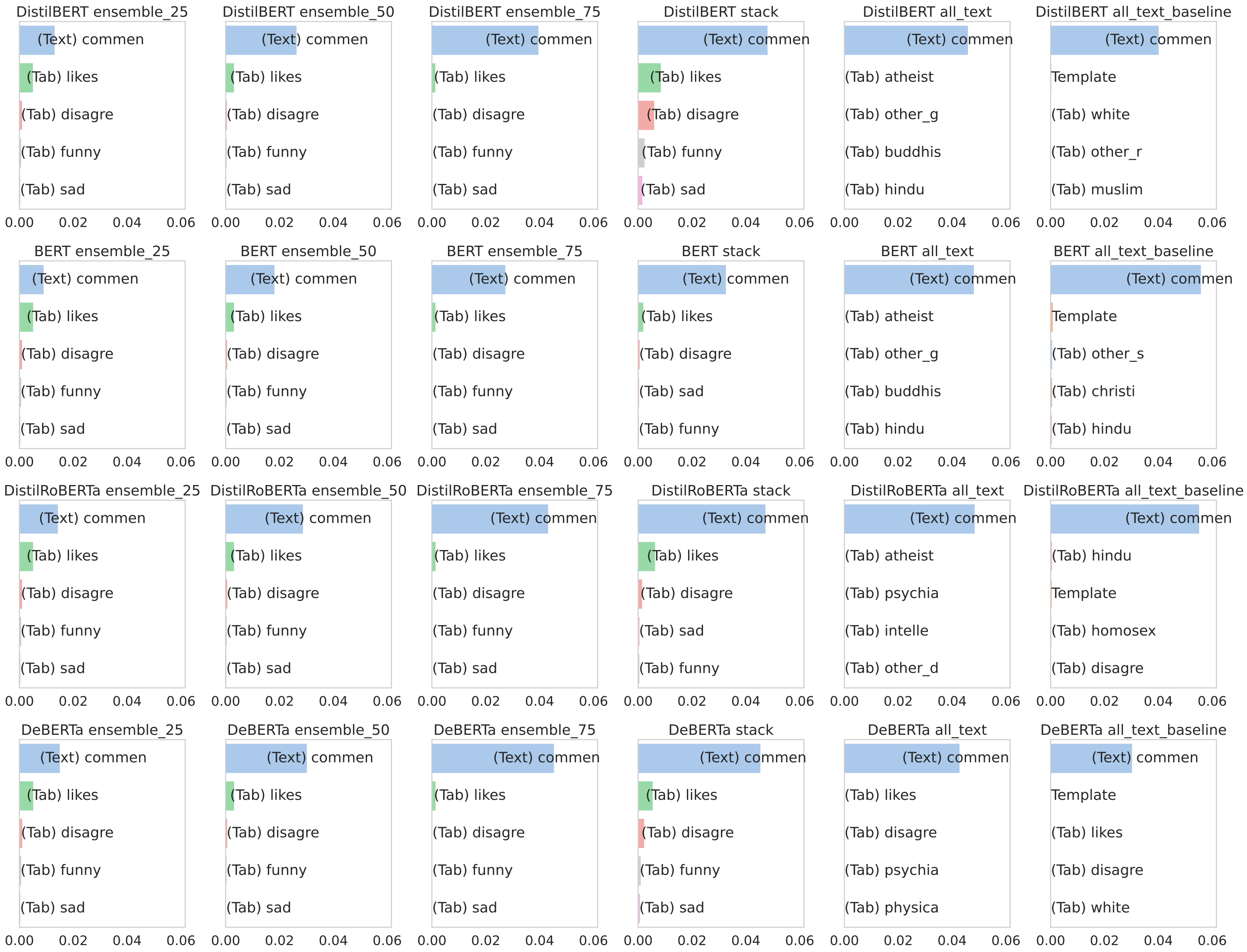

DistilBERT ensemble_25
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilBERT ensemble_50
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilBERT ensemble_75
(Text) descri
(Text) title
(Tab) require
(Text) salary
(Tab) require
DistilBERT stack
(Text) descri
(Tab) require
(Tab) require
(Text) title
(Text) salary
DistilBERT all_text
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilBERT all_text_baseline
(Text) descri
(Text) title
(Tab) require
Template
(Tab) require
BERT ensemble_25
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
BERT ensemble_50
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
BERT ensemble_75
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
BERT stack
(Tab) require
(Text) descri
(Text) title
(Tab) require
(Text) salary
BERT all_text
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
BERT all_text_baseline
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilRoBERTa ensemble_25
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilRoBERTa ensemble_50
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilRoBERTa ensemble_75
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilRoBERTa stack
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilRoBERTa all_text
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DistilRoBERTa all_text_baselin
(Text) descri
(Text) title
(Tab) require
(Tab) require
Template
DeBERTa ensemble_25
(Text) descri
(Tab) require
(Text) title
(Text) salary
(Tab) require
DeBERTa ensemble_50
(Text) descri
(Text) title
(Text) salary
(Tab) require
(Tab) require
DeBERTa ensemble_75
(Text) descri
(Text) title
(Text) salary
(Tab) require
(Tab) require
DeBERTa stack
(Text) descri
(Tab) require
(Tab) require
(Text) title
(Text) salary
DeBERTa all_text
(Text) descri
(Text) title
(Tab) require
(Tab) require
(Text) salary
DeBERTa all_text_baseline
(Text) descri
(Text) title
(Tab) require
Template
(Text) salary
0.00
0.25
0.50
0.75

(c) Fake



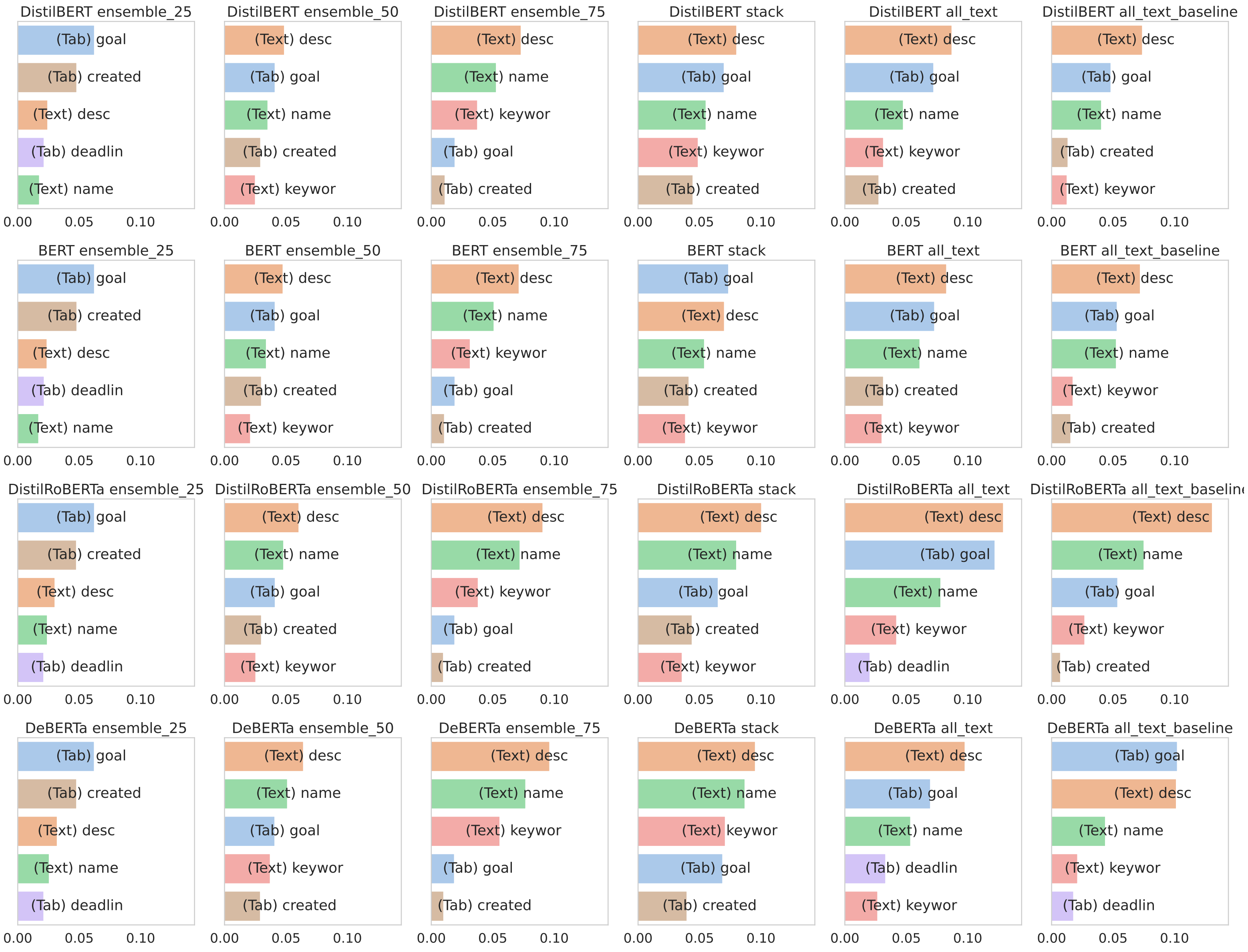
DistilBERT ensemble_25
(Tab) Rating
(Tab) Votes
(Tab) Revenue
(Tab) Runtime
(Text) Descri
DistilBERT ensemble_50
(Text) Descri
(Tab) Rating
(Tab) Votes
(Tab) Revenue
(Tab) Runtime
DistilBERT ensemble_75
(Text) Descri
(Tab) Rating
(Tab) Revenue
(Tab) Votes
(Tab) Runtime
DistilBERT stack
(Tab) Runtime
(Tab) Revenue
(Tab) Rating
(Tab) Rank
(Tab) Votes
DistilBERT all_text
(Text) Descri
(Tab) Votes
(Tab) Rank
(Tab) Revenue
(Tab) Runtime
DistilBERT all_text_baseline
(Text) Descri
Template
(Tab) Metasco
(Tab) Rank
(Tab) Rating
BERT ensemble_25
(Tab) Rating
(Tab) Votes
(Tab) Revenue
(Tab) Runtime
(Text) Descri
BERT ensemble_50
(Text) Descri
(Tab) Rating
(Tab) Votes
(Tab) Revenue
(Tab) Runtime
BERT ensemble_75
(Text) Descri
(Tab) Rating
(Tab) Revenue
(Tab) Votes
(Tab) Runtime
BERT stack
(Tab) Runtime
(Tab) Rating
(Tab) Rank
(Tab) Votes
(Tab) Revenue
BERT all_text
(Text) Descri
(Tab) Rank
(Tab) Votes
(Tab) Revenue
(Tab) Metasco
BERT all_text_baseline
(Text) Descri
(Tab) Metasco
Template
(Tab) Revenue
(Tab) Rank
DistilRoBERTa ensemble_25
(Tab) Rating
(Tab) Votes
(Tab) Revenue
(Tab) Runtime
(Text) Descri
DistilRoBERTa ensemble_50
(Text) Descri
(Tab) Rating
(Tab) Votes
(Tab) Revenue
(Tab) Runtime
DistilRoBERTa ensemble_75
(Text) Descri
(Tab) Rating
(Tab) Revenue
(Tab) Votes
(Tab) Runtime
DistilRoBERTa stack
(Tab) Runtime
(Tab) Rating
(Tab) Revenue
(Tab) Votes
(Tab) Rank
DistilRoBERTa all_text
(Text) Descri
(Tab) Year
(Tab) Rank
(Tab) Revenue
(Tab) Votes
DistilRoBERTa all_text_baseline
(Text) Descri
(Tab) Revenue
Template
(Tab) Metasco
(Tab) Votes
DeBERTa ensemble_25
(Tab) Rating
(Tab) Votes
(Tab) Revenue
(Tab) Runtime
(Text) Descri
DeBERTa ensemble_50
(Text) Descri
(Tab) Rating
(Tab) Votes
(Tab) Revenue
(Tab) Runtime
DeBERTa ensemble_75
(Text) Descri
(Tab) Rating
(Tab) Revenue
(Tab) Votes
(Tab) Runtime
DeBERTa stack
(Tab) Runtime
(Tab) Rating
(Tab) Revenue
(Tab) Rank
(Tab) Votes
DeBERTa all_text
(Text) Descri
(Tab) Rank
(Tab) Revenue
(Tab) Votes
(Tab) Runtime
DeBERTa all_text_baseline
(Text) Descri
(Tab) Rating
Template
(Tab) Metasco
(Tab) Rank
0.0
0.1
0.2
0.3

(d) Imdb

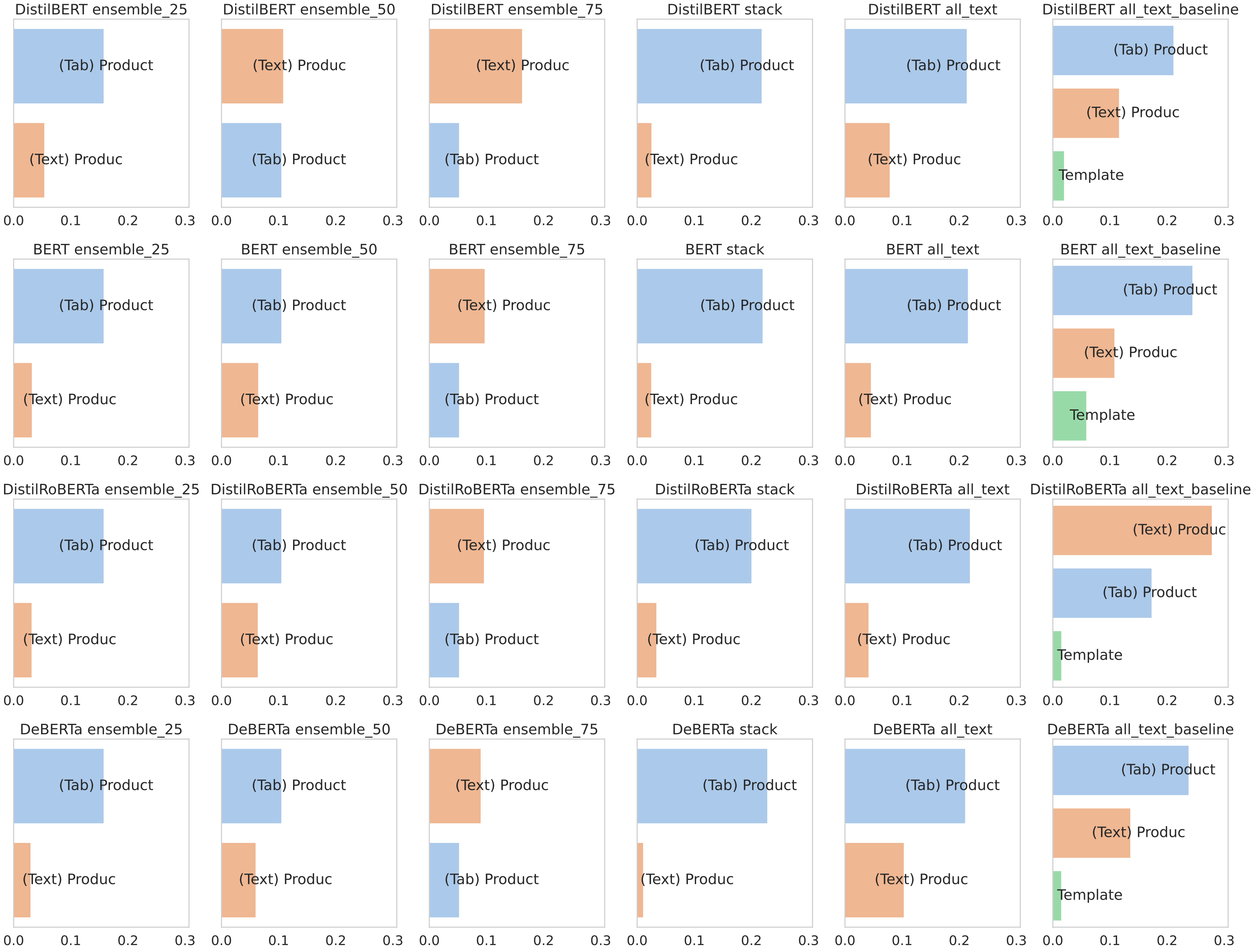


DistilBERT ensemble_25
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DistilBERT ensemble_50
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DistilBERT ensemble_75
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DistilBERT stack
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DistilBERT all_text
(Text) commen
(Tab) atheist
(Tab) other_g
(Tab) buddhis
(Tab) hindu
DistilBERT all_text_baseline
(Text) commen
Template
(Tab) white
(Tab) other_r
(Tab) muslim
BERT ensemble_25
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
BERT ensemble_50
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
BERT ensemble_75
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
BERT stack
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) sad
(Tab) funny
BERT all_text
(Text) commen
(Tab) atheist
(Tab) other_g
(Tab) buddhis
(Tab) hindu
BERT all_text_baseline
(Text) commen
Template
(Tab) other_s
(Tab) christi
(Tab) hindu
DistilRoBERTa ensemble_25
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DistilRoBERTa ensemble_50
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DistilRoBERTa ensemble_75
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DistilRoBERTa stack
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) sad
(Tab) funny
DistilRoBERTa all_text
(Text) commen
(Tab) atheist
(Tab) psychia
(Tab) intelle
(Tab) other_d
DistilRoBERTa all_text_baseline
(Text) commen
(Tab) hindu
Template
(Tab) homosex
(Tab) disagre
DeBERTa ensemble_25
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DeBERTa ensemble_50
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DeBERTa ensemble_75
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DeBERTa stack
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) funny
(Tab) sad
DeBERTa all_text
(Text) commen
(Tab) likes
(Tab) disagre
(Tab) psychia
(Tab) physica
DeBERTa all_text_baseline
(Text) commen
Template
(Tab) likes
(Tab) disagre
(Tab) white
0.00
0.02
0.04
0.06

(e) Jigsaw

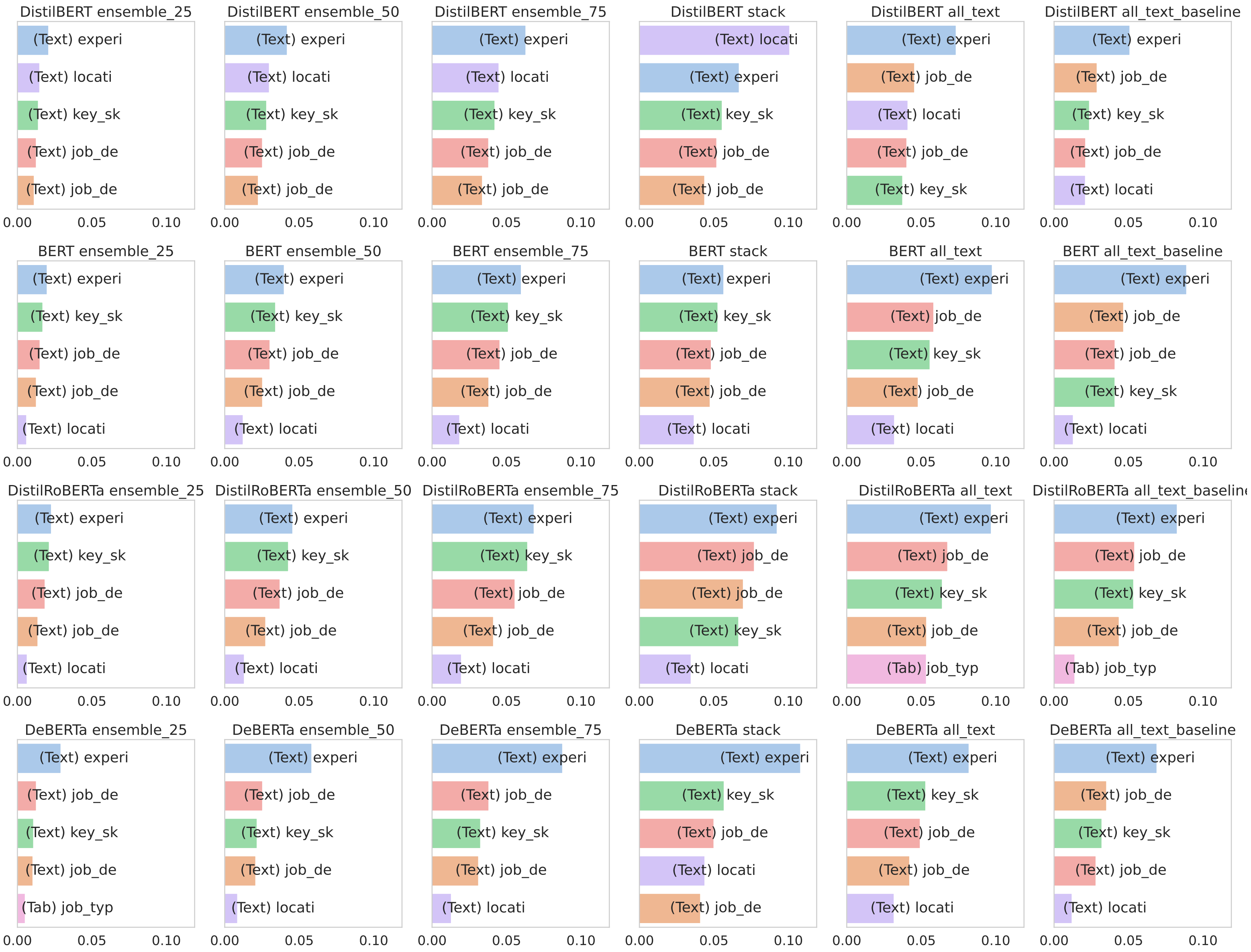

DistilBERT ensemble_25
(Tab) goal
(Tab) created
(Text) desc
(Tab) deadlin
(Text) name
DistilBERT ensemble_50
(Text) desc
(Tab) goal
(Text) name
(Tab) created
(Text) keywor
DistilBERT ensemble_75
(Text) desc
(Text) name
(Text) keywor
(Tab) goal
(Tab) created
DistilBERT stack
(Text) desc
(Tab) goal
(Text) name
(Text) keywor
(Tab) created
DistilBERT all_text
(Text) desc
(Tab) goal
(Text) name
(Text) keywor
(Tab) created
DistilBERT all_text_baseline
(Text) desc
(Tab) goal
(Text) name
(Tab) created
(Text) keywor
BERT ensemble_25
(Tab) goal
(Tab) created
(Text) desc
(Tab) deadlin
(Text) name
BERT ensemble_50
(Text) desc
(Tab) goal
(Text) name
(Tab) created
(Text) keywor
BERT ensemble_75
(Text) desc
(Text) name
(Text) keywor
(Tab) goal
(Tab) created
BERT stack
(Tab) goal
(Text) desc
(Text) name
(Tab) created
(Text) keywor
BERT all_text
(Text) desc
(Tab) goal
(Text) name
(Tab) created
(Text) keywor
BERT all_text_baseline
(Text) desc
(Tab) goal
(Text) name
(Text) keywor
(Tab) created
DistilRoBERTa ensemble_25
(Tab) goal
(Tab) created
(Text) desc
(Text) name
(Tab) deadlin
DistilRoBERTa ensemble_50
(Text) desc
(Text) name
(Tab) goal
(Tab) created
(Text) keywor
DistilRoBERTa ensemble_75
(Text) desc
(Text) name
(Text) keywor
(Tab) goal
(Tab) created
DistilRoBERTa stack
(Text) desc
(Text) name
(Tab) goal
(Tab) created
(Text) keywor
DistilRoBERTa all_text
(Text) desc
(Tab) goal
(Text) name
(Text) keywor
(Tab) deadlin
DistilRoBERTa all_text_baseline
(Text) desc
(Text) name
(Tab) goal
(Text) keywor
(Tab) created
DeBERTa ensemble_25
(Tab) goal
(Tab) created
(Text) desc
(Text) name
(Tab) deadlin
DeBERTa ensemble_50
(Text) desc
(Text) name
(Tab) goal
(Text) keywor
(Tab) created
DeBERTa ensemble_75
(Text) desc
(Text) name
(Text) keywor
(Tab) goal
(Tab) created
DeBERTa stack
(Text) desc
(Text) name
(Text) keywor
(Tab) goal
(Tab) created
DeBERTa all_text
(Text) desc
(Tab) goal
(Text) name
(Tab) deadlin
(Text) keywor
DeBERTa all_text_baseline
(Tab) goal
(Text) desc
(Text) name
(Text) keywor
(Tab) deadlin
0.00
0.05
0.10

(f) Kick

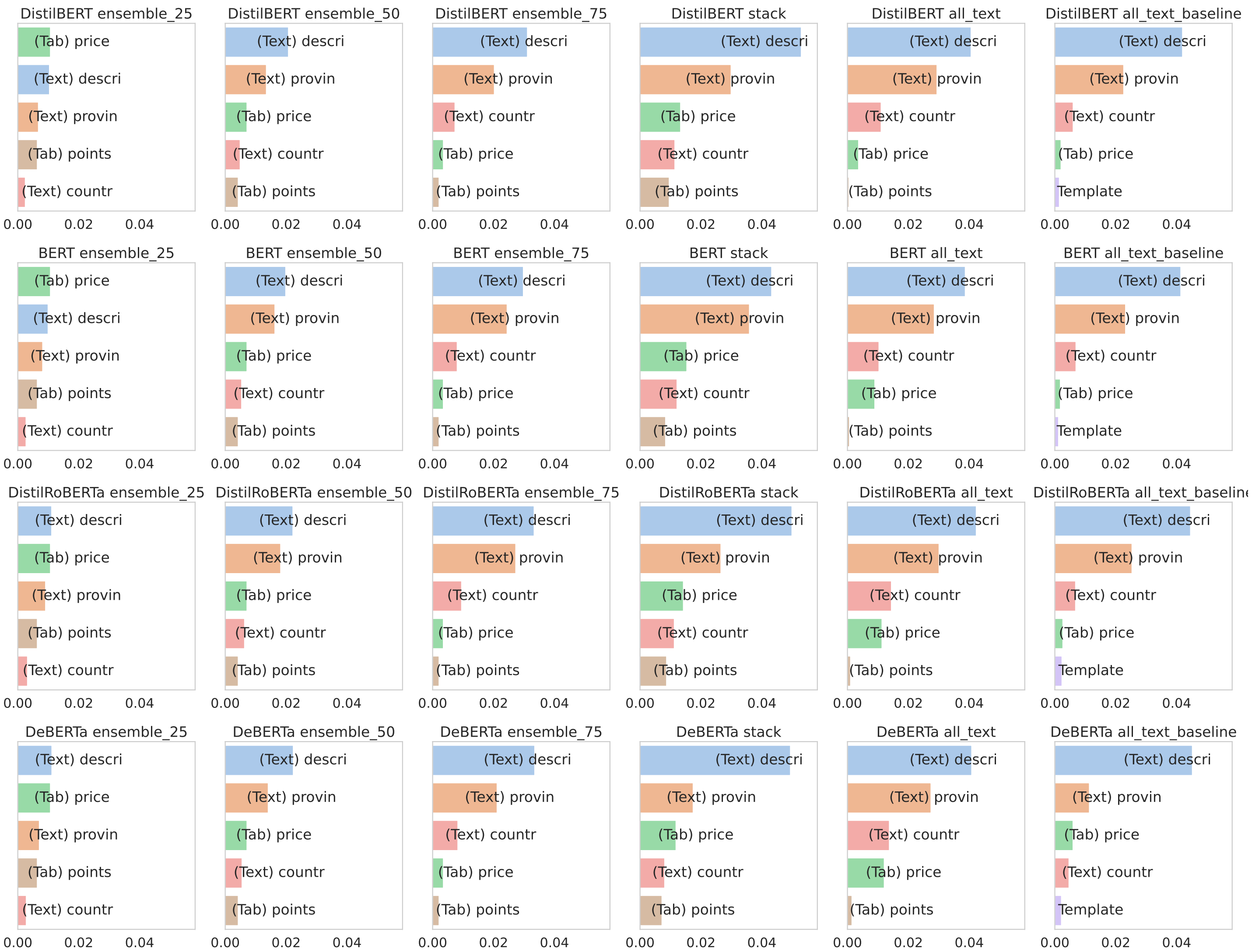


DistilBERT ensemble_25
(Tab) Product
(Text) Produc
DistilBERT ensemble_50
(Text) Produc
(Tab) Product
DistilBERT ensemble_75
(Text) Produc
(Tab) Product
DistilBERT stack
(Tab) Product
(Text) Produc
DistilBERT all_text
(Tab) Product
(Text) Produc
DistilBERT all_text_baseline
(Tab) Product
(Text) Produc
Template
BERT ensemble_25
(Tab) Product
(Text) Produc
BERT ensemble_50
(Tab) Product
(Text) Produc
BERT ensemble_75
(Text) Produc
(Tab) Product
BERT stack
(Tab) Product
(Text) Produc
BERT all_text
(Tab) Product
(Text) Produc
BERT all_text_baseline
(Tab) Product
(Text) Produc
Template
DistilRoBERTa ensemble_25
(Tab) Product
(Text) Produc
DistilRoBERTa ensemble_50
(Tab) Product
(Text) Produc
DistilRoBERTa ensemble_75
(Text) Produc
(Tab) Product
DistilRoBERTa stack
(Tab) Product
(Text) Produc
DistilRoBERTa all_text
(Tab) Product
(Text) Produc
DistilRoBERTa all_text_baseline
(Text) Produc
(Tab) Product
Template
DeBERTa ensemble_25
(Tab) Product
(Text) Produc
DeBERTa ensemble_50
(Tab) Product
(Text) Produc
DeBERTa ensemble_75
(Text) Produc
(Tab) Product
DeBERTa stack
(Tab) Product
(Text) Produc
DeBERTa all_text
(Tab) Product
(Text) Produc
DeBERTa all_text_baseline
(Tab) Product
(Text) Produc
Template
0.0
0.1
0.2
0.3

(g) Prod


DistilBERT ensemble_25
(Text) experi
(Text) locati
(Text) key_sk
(Text) job_de
(Text) job_de
DistilBERT ensemble_50
(Text) experi
(Text) locati
(Text) key_sk
(Text) job_de
(Text) job_de
DistilBERT ensemble_75
(Text) experi
(Text) locati
(Text) key_sk
(Text) job_de
(Text) job_de
DistilBERT stack
(Text) locati
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
DistilBERT all_text
(Text) experi
(Text) job_de
(Text) locati
(Text) job_de
(Text) key_sk
DistilBERT all_text_baseline
(Text) experi
(Text) job_de
(Text) key_sk
(Text) job_de
(Text) locati
BERT ensemble_25
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
(Text) locati
BERT ensemble_50
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
(Text) locati
BERT ensemble_75
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
(Text) locati
BERT stack
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
(Text) locati
BERT all_text
(Text) experi
(Text) job_de
(Text) key_sk
(Text) job_de
(Text) locati
BERT all_text_baseline
(Text) experi
(Text) job_de
(Text) job_de
(Text) key_sk
(Text) locati
DistilRoBERTa ensemble_25
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
(Text) locati
DistilRoBERTa ensemble_50
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
(Text) locati
DistilRoBERTa ensemble_75
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
(Text) locati
DistilRoBERTa stack
(Text) experi
(Text) job_de
(Text) job_de
(Text) key_sk
(Text) locati
DistilRoBERTa all_text
(Text) experi
(Text) job_de
(Text) key_sk
(Text) job_de
(Tab) job_typ
DistilRoBERTa all_text_baseline
(Text) experi
(Text) job_de
(Text) key_sk
(Text) job_de
(Tab) job_typ
DeBERTa ensemble_25
(Text) experi
(Text) job_de
(Text) key_sk
(Text) job_de
(Tab) job_typ
DeBERTa ensemble_50
(Text) experi
(Text) job_de
(Text) key_sk
(Text) job_de
(Text) locati
DeBERTa ensemble_75
(Text) experi
(Text) job_de
(Text) key_sk
(Text) job_de
(Text) locati
DeBERTa stack
(Text) experi
(Text) key_sk
(Text) job_de
(Text) locati
(Text) job_de
DeBERTa all_text
(Text) experi
(Text) key_sk
(Text) job_de
(Text) job_de
(Text) locati
DeBERTa all_text_baseline
(Text) experi
(Text) job_de
(Text) key_sk
(Text) job_de
(Text) locati
0.00
0.05
0.10

(h) Salary



(i) Wine

Figure 5.9: Top five features for each of the datasets, combination methods and text models.

5.5.4.2 Kendall's Tau

For each experiment, we compare the order of features, ranked by ϕ from most to least important, to allow comparisons across different experiments. Therefore, we chose to use Kendall's rank correlation coefficient, or Kendall's τ (Kendall, 1938), a non-parametric test to measure rank correlation. It is scored between -1 and 1, with identical rankings scoring 1 and opposite scoring -1.

$$\tau = \frac{\text{Number of concordant pairs} - \text{Number of discordant pairs}}{\text{Total number of pairs}} \tag{5.2}$$

For each TM-DS, for each of the 100 instances, we compare the ordering of features ranked by ϕ by calculating Kendall's τ between each pairing of CM and summarise the results in Table 5.3. The non-summarised results can be found in Table 8.1. We find that the three *Weighted-Ensembles* are the most similar to each other, as expected with their shared methodology. On the other hand, *All-Text* and *All-Text (Unimodal)* are most dissimilar to those with the most focus on tabular features: *Stack-Ensembles*. In general, *All-Text* is more similar to the remaining CMs than *All-Text (Uni)*.

Similarly, for each CM-DS, for each of the 100 instances, we calculate Kendall's τ between each pairing of TM and summarise the results in Table 5.4. The non-summarised results can be found in Table 8.2. One can see that changing TM causes less change in ϕ rankings than changing CM, with all pairs scoring a mean τ of 0.69-0.75. DeBERTa is the most different of the text models by a slight amount, likely because of the different way that it tokenizes, with spaces treated as part of the tokens.

Table 5.3: Mean (SD) Kendall's τ comparing the ϕ rankings of each combination method. Self-comparisons are trivially perfectly correlated ($\tau = 1$) and are omitted. Each result in is an average of n=3600 τ's (100 instances by 9 DSs by 4 TMs). Note for each excluded experiment, n will be less by 100. WE refers to *Weighted-Ensemble* and w indicates the weighting of the text model. AT refers to *All-Text* and AT (Uni.) refers to the unimodal version of *All-Text*.

	AT (Uni.)	AT	WE (w=.25)	WE (w=.50)	WE (w=.75)
All-Text	.46 (.32)				
WE (w=.25)	.30 (.33)	.36 (.38)			
WE (w=.50)	.38 (.34)	.45 (.37)	.83 (.17)		
WE (w=.75)	.40 (.34)	.46 (.37)	.69 (.24)	.85 (.16)	
Stack	.18 (.32)	.25 (.43)	.63 (.25)	.62 (.29)	.55 (.31)

Table 5.4: Mean (SD) Kendall's τ comparing the ϕ rankings of each text model. Self-comparisons are trivially perfectly correlated ($\tau = 1$) and are omitted. Each result is an average of n=5400 τ's (100 instances by 9 DSs by 6 CMs). Note for each excluded experiment, n will be less by 100.

	DistilBERT	BERT	DistilRoBERTa
BERT	.75 (.27)		
DistilRoBERTa	.73 (.29)	.71 (.29)	
DeBERTa	.70 (.28)	.69 (.28)	.69 (.30)

5.5.5 All Feature Analysis: Are Text and Tabular Features Weighted Differently?

Furthermore, we investigate whether the weighting of text and tabular features differ depending on the TM or CM used. As there is no guarantee of equal numbers of each feature type, we evaluate the difference between the average text ϕ and tabular ϕ, referring to this difference as Δ. We choose the median as ϕ is not normally distributed across instances.

$$\Delta = \phi_{\text{Median Text Feature}} - \phi_{\text{Median Tabular Feature}} \tag{5.3}$$

For each DS-TM combination, we test to see whether Δ differed between CMs. Similarly, for each DS-CM combination, we test to see whether Δ differed between TMs. Δ follows a non-normal distribution so we use the Kruskal-Wallis test (Kruskal and

Wallis, 1952). The Kruskal-Wallis test, otherwise known as a one-way ANOVA on ranks, is a non-parametric test for hypothesis testing between subjects for a non-normal continuous variable. The alternative hypothesis is that not all population medians are equal. $\hat{\varepsilon}^2_{\text{ordinal}}$ is the effect size of the Kruskal-Wallis test and indicates the percentage of the variation in the dependent variable that is explained by the independent variable.

We choose to focus here on one experiment, although similar charts for other experiments can be found in Figures 8.1 and 8.2. In Figure 5.10, we compare Δ's for each CM when DS=*fake* and TM=BERT. The significant Kruskal-Wallis test$\chi^2_{\text{Kruskal Wallis}}(5) = 241.30, p = 4.03e - 50$ indicates that all medians are not equal, as is clear visually. Additionally, $\hat{\varepsilon}^2_{\text{ordinal}} = 0.40$ indicates that 40% of the variation in Δ is explained by the changes in the combination method. Figure 5.10 also shows non-significant pairwise tests, indicated with a bar. As Kruskal-Wallis tests if a group of populations are the same, in order to compare two groups, Dunn's test (Dunn, 1961) is appropriate. These tests have the Holm multiple-comparison adjustment applied.

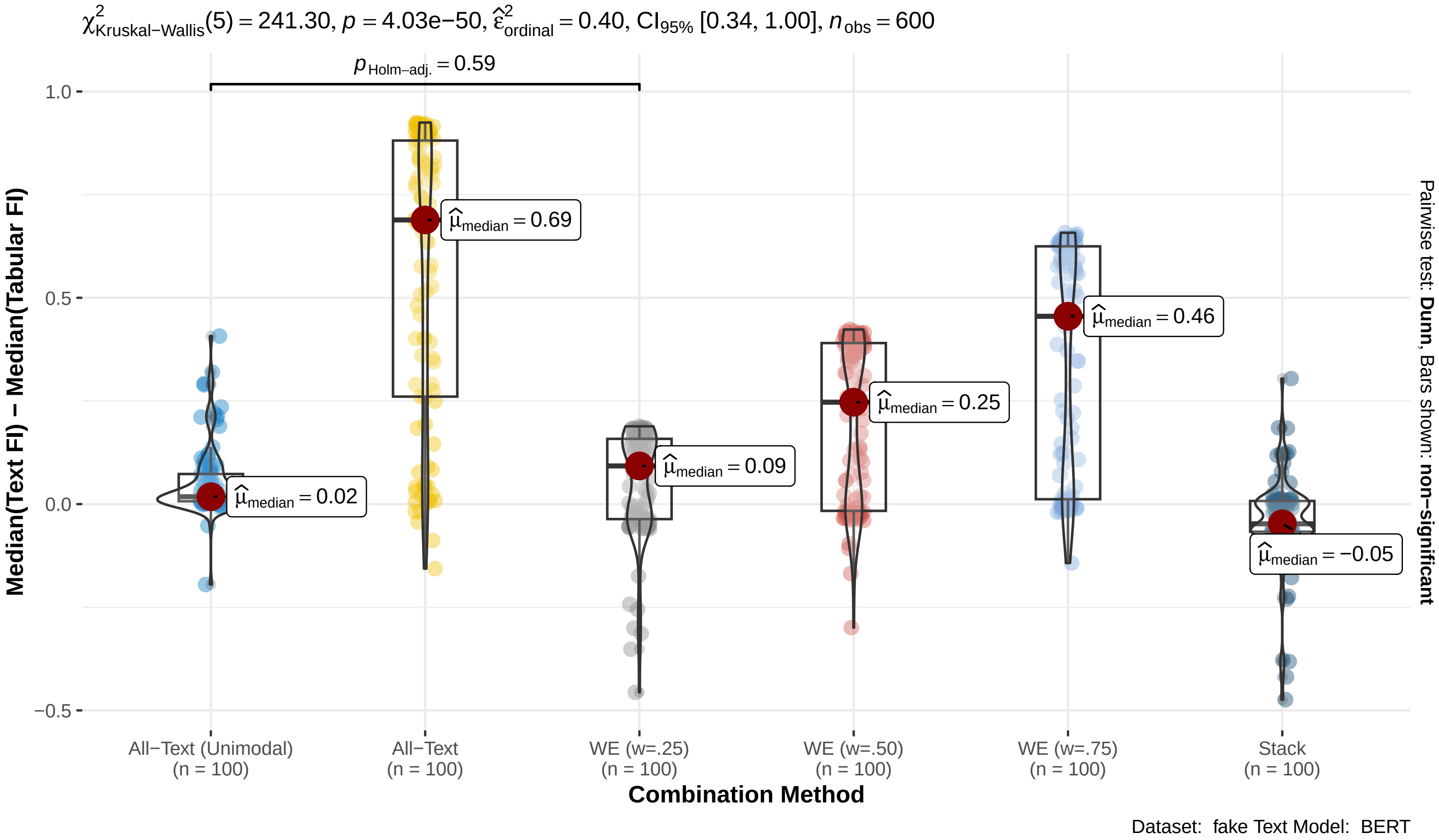


Figure 5.10: In order to compare which combination method assigns more relative importance to text or tabular features, we plot $\Delta = \phi_{\text{Median Text Feature}} - \phi_{\text{Median Tabular Feature}}$on the y-axis and combination method on the x-axis. This example sets dataset=*fake* and text model=BERT. Each differently coloured plot represents the distribution of Δ's for a single combination method, each with 100 coloured dots representing Δ for each of the 100 instances. For each plot, the distribution is also represented by a violin plot, which emphasises the non-normal distributions, a box plot, which indicates the spread of the data, and a labelled red dot, which indicates the median. A higher Δ indicates that a higher relative importance is assigned to text features over tabular features. The Kruskal-Wallis test statistic is significant ($\chi^2_{\text{Kruskal Wallis}}(5) = 241.30, p = 4.03e-50$), meaning that the alternative hypothesis of all medians not being equal is accepted. Of the pairwise tests, only [WE (w=.25), *All-Text*] is non-significant. WE refers to *Weighted-Ensemble* and w indicates the weighting of the text model.

Considering $\hat{\varepsilon}^2_{\text{ordinal}}$ for each experiment, we report the results in Tables 5.5 and 5.6. Table 5.5 shows the proportion of variance in Δ explained by the changes in combination method for each of the four text models. Table 5.6 shows the proportion of variance in Δ explained by the changes in the text model for each of the six combination methods.

Table 5.5: Effect size by model. We report the Mean (SD) effect size of the Kruskal-Wallis test ($\hat{\varepsilon}^2_{\text{ordinal}}$) when we test to see what proportion of the variance in the dependent variable, Δ, can be explained by changes in the independent variable: combination method.

	airbnb	channel	fake	imdb	jigsaw	kick	prod	salary	wine
BERT	0.46	0.53	0.40	0.66	0.40	0.23	0.54	0.31	0.56
DeBERTa	0.47	0.52	0.45	0.50	0.46	0.33	0.56	0.29	0.48
DistilBERT	0.46	0.53	0.46	0.53	0.44	0.25	0.67	0.40	0.66
DistilRoBERTa	0.45	0.48	0.43	0.57	0.29	0.23	0.74	0.37	0.60
Mean (SD)	.46 (.01)	.52 (.02)	.44 (.02)	.57 (.06)	.40 (.07)	.26 (.04)	.63 (.08)	.34 (.04)	.58 (.07)

Table 5.6: Effect size by method. We report the Mean (SD) effect size of the Kruskal-Wallis test ($\hat{\varepsilon}^2_{\text{ordinal}}$) when we test to see what proportion of the variance in the dependent variable, Δ, can be explained by changes in the independent variable: text model.

	airbnb	channel	fake	imdb	jigsaw	kick	prod	salary	wine
All-Text	0.19	NaN	0.28	0.06	0.18	0.09	0.40	0.24	0.04
All-Text (Uni.)	0.06	NaN	0.33	0.09	0.36	0.09	0.18	0.05	0.12
WE (w=.25)	0.12	0.08	0.21	0.00	0.35	0.05	0.06	0.05	0.04
WE (w=.50)	0.13	0.13	0.26	0.00	0.35	0.07	0.16	0.06	0.11
WE (w=.75)	0.13	0.14	0.28	0.01	0.35	0.08	0.16	0.07	0.14
Stack	0.03	0.23	0.09	0.11	0.30	0.10	0.15	NaN	NaN
Mean (SD)	.11 (.05)	.15 (.05)	.24 (.08)	.05 (.04)	.32 (.06)	.08 (.02)	.19 (.10)	.09 (.07)	.09 (.04)

Table 5.7: To represent a single experiment, we take the median Δ and report the Mean (SD) for each Combination Method. Higher values of Δ indicate a higher weighting of text features compared to tabular features. The median Δ for an experiment represents 100 instances. Each result is an average of n=36 experiments (4 TMs by 9 DSs) and w indicates the weighting of the text model.

Combination Method	Median Δ
All-Text (Unimodal)	.08 (.29)
All-Text	.15 (.37)
Weighted Ensemble (w=.25)	-.03 (.16)
Weighted Ensemble (w=.50)	.09 (.14)
Weighted Ensemble (w=.75)	.22 (.19)
Stack Ensemble	-.06 (.23)

Taking the median Δ from each of the 199 experiments (the labelled dark red circles

Table 5.8: To represent a single experiment, we take the median Δ and report the Mean (SD) for each Text Model. Higher values of Δ indicate a higher weighting of text features compared to tabular features. The median Δ for an experiment represents 100 instances. Each result is an average of n=54 experiments (6 CMs by 9 DSs).

Text Model	Median Δ
BERT	.05 (.29)
DistilBERT	.10 (.21)
DistilRoBERTa	.10 (.28)
DeBERTa	.06 (.25)

in Figure 5.10 show six of them), we average over text model and combination method and report the summary statistics in Tables 5.7 and 5.8 respectively. We see that changing the text model has much less of an impact than changing the combination method. Looking at Table 5.7, the three *Weighted-Ensembles* follow a predictable pattern, with the preference for text over tabular features as w increases from 0.25 through to 0.75. Specifically, we see Δ increasing as the weighting of the text model predictions increases. When comparing Δ for *All-Text* and *All-Text (Unimodal)*, we observe that unimodal explanations put a higher weight on tabular features (reflecting a lower Δ). *Stack-Ensembles*, on average, placed the lowest relative emphasis on text features. We hypothesise this to be a consequence of tabular features appearing twice in the training process, first at the initiation of the tabular model and then during the meta-model.

5.5.6 All Feature Analysis: Explainability Consistency

In order to compare the quality of our multimodal masker with the unimodal masker, we first point out that both retain all the qualities of a SHAP explanation, namely local accuracy, missingness and consistency. However, another method of explanation quality was proposed in Watson et al. (2022), which looked at how consistent entire explanations were when regenerating SHAP values after retraining the same model architecture with a different random seed. A simple linear model, M, is trained to classify between the SHAP values of models a and b, and α is the number of comparisons made. The metric is scaled to be between 0 and 1, with a perfectly confused linear model with 50% accuracy scoring 1.

$$C = 1 - \frac{\Sigma_{(a,b)} 2 * |M(a,b) - 0.5|}{\alpha} \tag{5.4}$$

We choose to experiment with TM=DistilRoBERTa and DS=*fake* for strong performance and a similar number of text and tabular features. We finetune the text model four times with four different random seeds, then generate SHAP values from the same 1000 test set instances using both maskers using CM=*All-Text*. For each masker, a linear model is trained with 10-fold cross-validation to distinguish each of the six unique pairings ($\alpha = 6$). Table 5.9 shows the results.

Table 5.9: Using CM=*All-Text*, TM=DistilRoBERTa and DS=*fake*, we compare explanation consistency when using multimodal and unimodal maskers.

Combination Method	Explanation Consistency
All-Text (Unimodal)	.659
All-Text	.853

5.5.7 Reordering and Retraining Based on Feature Importance Order

Here, we aim to test whether the order of features can be manipulated to achieve greater performance on the test set. For each experiment, we reorder the features in the *All-Text* template based on how important each feature was ranked. For each experiment, we train two new models: one based on how features were ranked using the multimodal masker and one based on how features were ranked using the unimodal masker. The newly ordered features are fit to the string template from most to least important. We report the results in Table 5.10.

Assessing the results, there is no clear improvement from reordering the features. The differences between the two new models and the original are small with no clearly best-performing strategy. Therefore, we cannot say that reordering makes a significant difference, nor that using the unimodal masker rankings over multimodal - or vice versa - provided a significant change in performance.

Table 5.10: Model results based on models retrained with inputs reordered from most to least important. The ranking of features and subsequent retraining was done twice, first from *All-Text (Unimodal)*, (results under UM Reorder) and second from *All-Text*, which uses the multimodal masker (results under MM Reorder). We also report the original score from Table 5.2.

		airbnb (Acc.)	channel (Acc.)	fake (ROC)	imdb_genre (ROC)	jigsaw (ROC)	kick (ROC)	prod_sent (Acc.)	salary (Acc.)	wine (Acc.)
BERT	Original	**.387**	**.254**	**.962**	.828	.961	.781	**.905**	**.481**	.826
	UM Reorder	.378	**.254**	.955	.804	**.962**	.521	**.905**	.469	.826
	MM Reorder	.374	**.254**	.941	**.837**	.957	**.788**	**.905**	.473	**.829**
DeBERTa	Original	.377	.317	**.959**	**.797**	.955	.776	**.888**	.458	.817
	UM Reorder	.378	.254	.955	.767	**.962**	.789	**.888**	.451	**.821**
	MM Reorder	**.381**	**.332**	.956	.777	.957	**.792**	**.888**	**.459**	.819
DistilBERT	Original	**.380**	**.319**	**.961**	**.815**	.962	**.788**	**.901**	.458	**.819**
	UM Reorder	.358	**.319**	**.961**	**.815**	**.964**	.784	**.901**	.463	**.819**
	MM Reorder	.377	.317	.958	.813	.963	.786	**.901**	**.471**	.816
DistilRoBERTa	Original	**.390**	.313	.958	**.772**	**.964**	**.795**	.904	**.471**	.821
	UM Reorder	.366	**.321**	.958	.760	.958	.790	**.907**	**.471**	.821
	MM Reorder	.371	.299	**.962**	.760	.959	.789	.904	.470	**.826**

5.6 Conclusion

In this study, we enable the generation of SHAP values for any text-tabular combination method for the first time. This novel multimodal masker facilitates the masking of text and tabular features without first requiring conversion into a single modality. This opens up new avenues for explainability in multimodal models, allowing one to gain insights into new combination methods for the first time. We addressed the limitations of the existing unimodal masker, which restricted the generation of SHAP explanations to the All-Text combination method. By deferring input formation until after the masking stage and treating text and tabular features consistently regardless of how they are combined, one can avoid the pitfalls of the unimodal approach.

Through extensive experimentation across nine datasets, we explored how changing text models and combination methods affect the resulting explanations. In particular, we find *All-Text* models favour textual features, whereas *Weighted-Ensembles* with a low text weighting w, and *Stack-Ensembles* - which see each tabular feature twice - favour tabular features. Across all datasets, changing the combination method had a greater impact on feature importance rankings than changing the text model. Finally, we retrained and regenerated SHAP values for CM=*All-Text*, TM=DistilRoBERTa, and DS=*fake* and found that the multimodal masker produces more consistent explanations than the unimodal masker.

Although we experimented with many datasets, text models, and several combination methods, there are still other factors that were omitted and could be experimented with. For *All-Text*, only a single style of string template was used. Future work could explore how explanations are affected when templates are varied. Furthermore, this work focuses on the initial implementation and subsequent analysis; further work could target speed and efficiency gains with specific configurations for certain model types and combination methods, as have been developed for the original SHAP library.

5.7 Additional Material

Chapter 6

Explainable Text-Tabular Models for Predicting Mortality Risk in Companion Animals

In this chapter, we apply the multimodal masking framework from the previous chapter (Chapter 5) to identify risk factors for companion animal mortality in first-opinion veterinary EHRs from across the United Kingdom. As clinical datasets contain a range of modalities, from the free-text of clinician notes to structured tabular data entries, there was a clear opportunity to apply the framework to provide comprehensive explanations. As in Chapter 5, we present five multimodality approaches, with the best-performing method utilising PetBERT, a language model pre-trained on a veterinary dataset (see Section 2.2).

we investigate the important features, demonstrating PetBERT's inclination to engage more with free-text narratives compared to BERT-base. The investigation also explores the important features on a more granular level, identifying distinct words and phrases that substantially influenced an animal's life status prediction. PetBERT showcased a heightened ability to grasp phrases associated with veterinary clinical nomenclature, signalling the productivity of additional pre-training of language models.

6.1 Introduction

Life expectancy serves as a fundamental metric for understanding human and animal populations' overall health and well-being (Roser et al., 2013). Understanding life expectancies permits insights into the health status of a populace and aids in the identification of health disparities and inequalities between specific regions. Tools designed for monitoring mortality play a vital role in assisting researchers in pinpointing events occurring earlier in life that may reduce overall lifespan. Nevertheless, national mortality rates for companion animals are not subject to regular monitoring. The surveillance of EHRs collected from primary-care veterinary practices represents a valuable means to gain insights into companion animals' current population health status and we discuss one particular dataset collected by the Small Animal Veterinary Surveillance Network (SAVSNET) in Section 2.2.

Despite their potential, it is challenging to harness the total utility of first-opinion veterinary EHRs on a large scale. The implementation of disease coding frameworks, while advantageous for researchers, often proves counter-intuitive in clinical practice and impractical for everyday use. Previous studies have underscored records annotated by clinicians as part of their routine responsibilities as being particularly susceptible to inaccuracies and omissions (Lloyd and Rissing, 1985; Zafirah et al., 2018). Adopting an unstructured, free-text format in contemporary veterinary EHRs while affording clinicians greater linguistic flexibility presents challenges in developing automated systems (Mcdonald, 1997; Rosenbloom et al., 2017).

Combining this free text field with tabular data points summing up their clinical history has been used by Farrell et al. (2023b) to train models predicting the mortality of companion animals in the SAVSNET dataset. However, the limited arsenal of explainability techniques means that current methods, such as SHAP (Lundberg and Lee, 2017a) or LIME (Ribeiro et al., 2016a), are insufficient, as they can only address one modality at a time. Consequently, a compelling need arises to develop comprehensive explainability systems for deep learning models in clinical contexts that capture all the available data and ensure that healthcare practitioners make clinical decisions with confidence and trust.

This chapter builds upon the previous research by applying the novel multimodal masking framework that was introduced in Chapter 5 to the veterinary clinical domain. We extend the work of Farrell et al. (2023b), applying five combination techniques of combining modalities for mortality prediction. Using the multimodal masking framework, we generate SHAP explanations for all techniques and investigate the reasons for performance differences, particularly the improvement when using an additionally pre-trained large language model (PetBERT) that is tailored to veterinary clinical records.

The investigation delves into the unique words, phrases and individual tabular values to directly compare which characteristics significantly affect the prediction of an animal's living status. By incorporating multiple modalities, including breed, age, deprivation scores, and clinical narratives, we unveil the features contributing to increased mortality risk. The implications of this research extend beyond animal welfare and highlight the potential of a multimodality explanation framework applicable across diverse tasks.

To summarise, the contributions of this chapter are as follows:

- The training of a selection of models across five combination types and two text models to predict companion animal mortality from a veterinary text-tabular dataset.
- An application of the multimodal masking framework proposed in the previous chapter to generate explanations for each of the methods and a subsequent analysis of the risk factors associated with animal mortality.
- A more granular investigation on the individual words, phrases and tabular values that are most influential for a model prediction.

6.2 Dataset

For the dataset, we start with the SAVSNET dataset, prepared for mortality prediction in Farrell et al. (2023b), using the semi-supervised mortality labels as prediction targets. Whereas the authors build a model for predicting mortality 1-12 months in advance, we look only at predicting mortality 1 month in advance so that our analysis can focus on the difference between combination methods and text models. Furthermore, we make adjustments as to which features are counted as text and tabular, treating categorical features with more than 30 unique values as text features, namely *breed* and *region*. This was done to avoid extensive one-hot encoding or ordinally encoding variables with no linear relationship.

6.3 Method

6.3.1 Model Training

we determine whether an animal known to have died within 28 days of the last given consultation can be identified using all five combination methods from the previous chapter, first with BERT as the language model and then repeat with PetBERT as the language model, for a total of 10 experiments. SAVSNET data contains a mixture of free-text and tabular features, so in this study, we utilise five different methods of training a model with both modalities. Figure 6.1 outlines the combination methods. As in Chapter 5, we use the *All-Text* approach, three *Weighted-Ensembles* models and a *Stack-Ensemble*. In the *All-Text* approach, all features are fit to a string template and fed to a large language model; the following format is utilised:

Column name 0: Column value 0 | Column name 1: Column value 1 | ...

For the *Weighted-Ensembles*, text and tabular models are trained separately, and their predictions are combined in weighted sum, with w as the weight of the text model prediction and *1-w* as the weight of the tabular model prediction). Once more, we experiment with three values of w: 0.25, 0.50 and 0.75. The *Stack-Ensemble* method also requires separately trained text and tabular models but also has a third model, a meta-model, trained on the tabular features and the text and tabular predictions using the validation set. All tabular and meta-models are light gradient boosting classifiers (Ke et al., 2017).

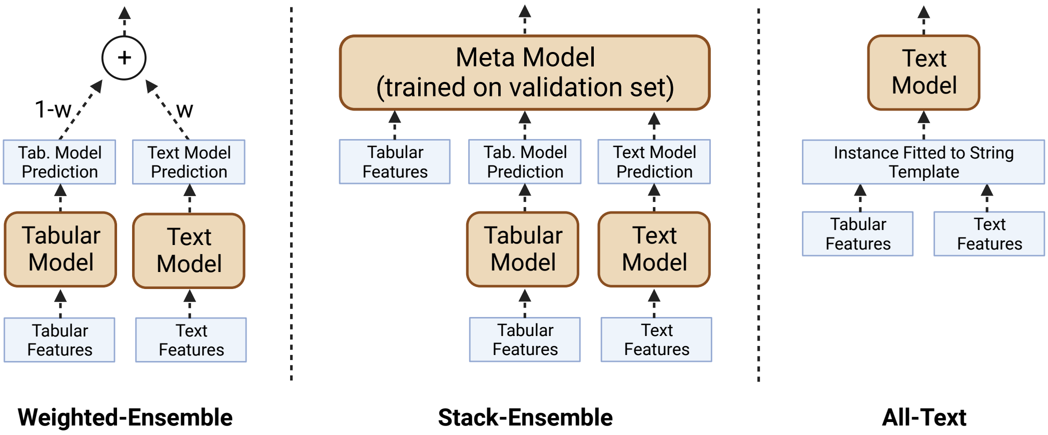


Figure 6.1: Combination methods used in this study, first seen in Chapter 5. **Left:** A *Weighted-Ensemble*. **Middle:** A *Stack-Ensemble*. **Right:** *All-Text*.

6.3.2 Model Evaluation

For each text model-combination method pair, we evaluate the performance against a test dataset selected from the 2m records set aside from the initial pre-training of PetBERT in Farrell et al. (2023b). Therefore, this test set contained records that had not been seen by PetBERT in either the initial masked learning step or in the downstream classifications step. Following the original PetBERT paper (Farrell et al., 2023b), we report performance using the F1 score, see Table 6.1a. For added information, we also report accuracy in Table 6.1b.

Table 6.1: Test set F1 (a) and accuracy (b) scores for all models. We also report the scores when only the text columns are used to train the language models, labelling this *All Text (Txt fts only)*. WE refers to *Weighted-Ensemble*

(a) F1 scores on test set

	PetBERT	BERT
Stack	.828	.821
WE w=.25	.822	.823
WE w=.50	**.844**	**.828**
WE w=.75	.828	.800
All Text	.831	.823
All Text (Txt fts only)	.811	.783
Tabular Model	.802	

(b) Accuracy scores on test set

	PetBERT	BERT
Stack	.820	.807
WE w=.25	.809	.810
WE w=.50	**.832**	**.817**
WE w=.75	.814	.785
All Text	.823	.816
All Text (Txt fts only)	.794	.766
Tabular Model	.789	

6.3.3 Generating SHAP values

The goal is to explore the reasons for the similarities and differences in the performances and investigate why PetBERT outperformed BERT for four of the five combination methods. To do so, we generate SHAP values for each combination of the two independent variables: combination method (CM) and text model (TM). To isolate the differences in explanations to the independent variables, we choose the same 1000 randomly selected test-set examples to be explained for each TM-CM combination.

6.4 Results

To account for the variations observed in label counts and token quantities across all instances, we utilise a process similar to those developed within the original SHAP package's summary plot function. Whereas tabular features produce a single SHAP value, text features produce a SHAP value for each word piece. Therefore, we sum the SHAP values such that each feature is represented by a single value. Specifically, there are T tokens for each instance, each belonging to one of F features. Each token has associated SHAP values for L labels, which for this binary classification task is 2. First, the SHAP values for each token are summed, $t \in T$, belonging to a feature, $f \in F$ before converting to the absolute value and sum across each of the two labels, $l \in$ [alive, dead]. Therefore, a single SHAP value for each feature in each instance indicates how important the feature was to the model. We refer to this as feature importance or ϕ.

$$\phi_f = |\sum_{t \in f} \text{SHAPvalue}_{t,l=\text{alive}}| + |\sum_{t \in f} \text{SHAPvalue}_{t,l=\text{dead}}| \tag{6.1}$$

Note that this is the same feature importance formula as the previous chapter. However, for clarity, the formula is written verbosely for *alive* and *dead* as there are only two labels and a single dataset, instead of a sum across all labels.

6.4.1 Which Features are the Most Important?

Typically, to see how important a feature is across the entire dataset, one would use a SHAP summary plot, which shows mean absolute SHAP values for each feature, averaged across the entire dataset. In this analysis, we use mean ϕ. To compare across experiments, for each TM-CM pair, we plot mean ϕ as a proportion of the sum of all mean absolute ϕ for that pair. This is shown in Figure 6.2.

For each of the three *Weighted-Ensembles*, one can see a linear increase in the reliance on textual features as w, the text model weighting, increases, a pattern seen for both BERT and PetBERT. When w=0.25, *age at consult* is the most important feature overall, whereas for w=0.50 and 0.75, *clinical narrative* is the most important. The *Stack-Ensembles* follow the pattern of PetBERT relying on *clinical narrative* more than BERT, however, with *age at consult* as the most influential feature. The *All-Text* models represent the only experiments where tabular features are fed into a text model. These results demonstrate that despite this, language models can indeed extract use out of tabular features with both PetBERT *All-Text* and BERT *All-Text* focusing on *age at consult* the most.

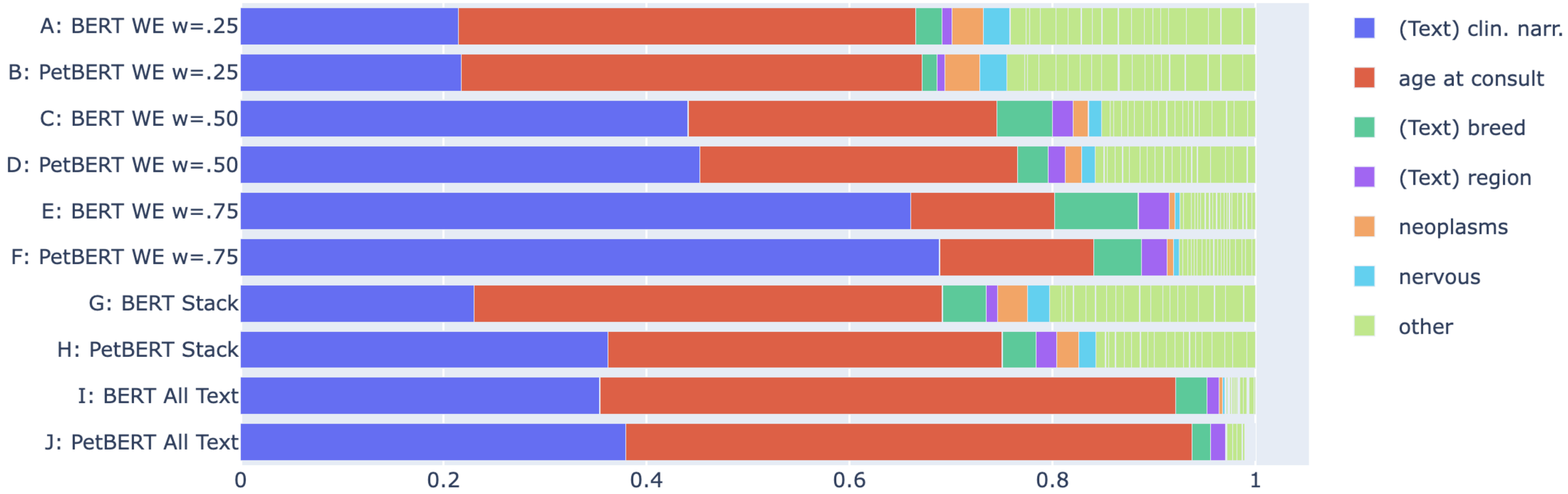


Figure 6.2: Mean feature importance, by feature, as a proportion of the sum of all mean absolute feature importance. Each row indicates the proportion for a particular model, as indicated on the y-axis. We look across all experiments to find the six most influential features, colouring the remaining features as lime-green in an *other* category. The size of each coloured bar indicates the feature's relative overall importance to the particular model, with cumulative proportion on the x-axis. The order of the colours is the same for each row, in order of highest to lowest proportion across all experiments.

6.4.2 How Similarly are Features Ranked?

In this section, we look at how statistically similar the rankings of features are for a given instance. To do so, we use Kendall's rank correlation coefficient(Kendall, 1938), or Kendall's τ. A description of Kendall's τ is given in Section 5.5.4.2.

In Figure 6.2, results were averaged across all instances and then reported. Here, we calculate Kendall's τ for each instance and then average, reporting the mean and standard deviation of the statistic to facilitate a more nuanced examination. In Tables 6.2, 6.3 and 6.4, for a particular comparison, we calculate τ between the two rankings for each of the 1000 instances and then report the mean and standard deviation of those 1000 scores in the table. In Table 6.2, for each of the five combination methods, we compare the similarity between rankings when using BERT versus PetBERT. In Table 6.3, we compare each of the five combination methods against each other when fixing TM=BERT, whereas in Table 6.4, we do the same but fix TM=PetBERT.

Following evidence in Figure 6.2 of both *All-Text* models focusing almost entirely on two features, one can see these models as the most dissimilar to other combination methods with scores between 0.43-0.52 for BERT (Table 6.3) and 0.31-0.41 for PetBERT (Table 6.4). Despite this, they are also dissimilar to each other with a mean τ of just 0.37 (Table 6.2), which suggests that even if the remaining features are similarly small in magnitude (from Figure 6.2), they are not often in a similar order. For both text models, the two most similar combination method pairs are [*Weighted-Ensemble w=.50*, *Weighted-Ensemble w=.25*] and [*Weighted-Ensemble w=.50*, *Weighted-Ensemble w=.75*]. With a shared methodology, similarity is expected; with only the weighting on the prediction changing, it will only be the ordering of the tabular features relative to the text features that differ. Table 6.2 shows how much of a difference changing text model has when fixing the combination method and shows high mean scores of 0.80-0.81 for each of the *Weighted-Ensembles*, indicating a similar ordering of features for a given instance.

Table 6.2: Mean (SD) Kendall's τ comparing the ϕ rankings of BERT vs PetBERT for each of the five combination methods. Self-comparisons are trivially perfectly correlated ($\tau = 1$) and are omitted. For each entry, we calculate Kendall's tau for each of the n=1000 instances and report the Mean (SD). WE refers to *Weighted-Ensemble* with w indicating text model weighting.

Method	Model A vs Model B	Mean (Std)
All-Text	BERT vs PetBERT	0.37 (0.23)
WE 25		0.81 (0.11)
WE 50		0.81 (0.14)
WE 75		0.80 (0.18)
Stack		0.70 (0.15)
Combined	BERT vs PetBERT	0.70 (0.24)

Table 6.3: Mean (SD) Kendall's τ comparing the ϕ rankings of each of the combination methods against each other, split by BERT models. Self-comparisons are trivially perfectly correlated ($\tau = 1$) and are omitted. For each entry, we calculate Kendall's tau for each of the n=1000 instances and report the Mean (SD). WE refers to *Weighted-Ensemble* with w indicating text model weighting.

BERT, by method	All-Text	WE 25	WE 50	WE 75
WE 25	0.43 (0.19)			
WE 50	0.50 (0.21)	0.73 (0.16)		
WE 75	0.52 (0.24)	0.56 (0.20)	0.70 (0.19)	
Stack	0.45 (0.19)	0.71 (0.14)	0.64 (0.17)	0.53 (0.20)

Table 6.4: Mean (SD) Kendall's τ comparing the ϕ rankings of each of the combination methods against each other, split by PetBERT models. Self-comparisons are trivially perfectly correlated ($\tau = 1$) and are omitted. For each entry, we calculate Kendall's tau for each of the n=1000 instances and report the Mean (SD). WE refers to *Weighted-Ensemble* with w indicating text model weighting.

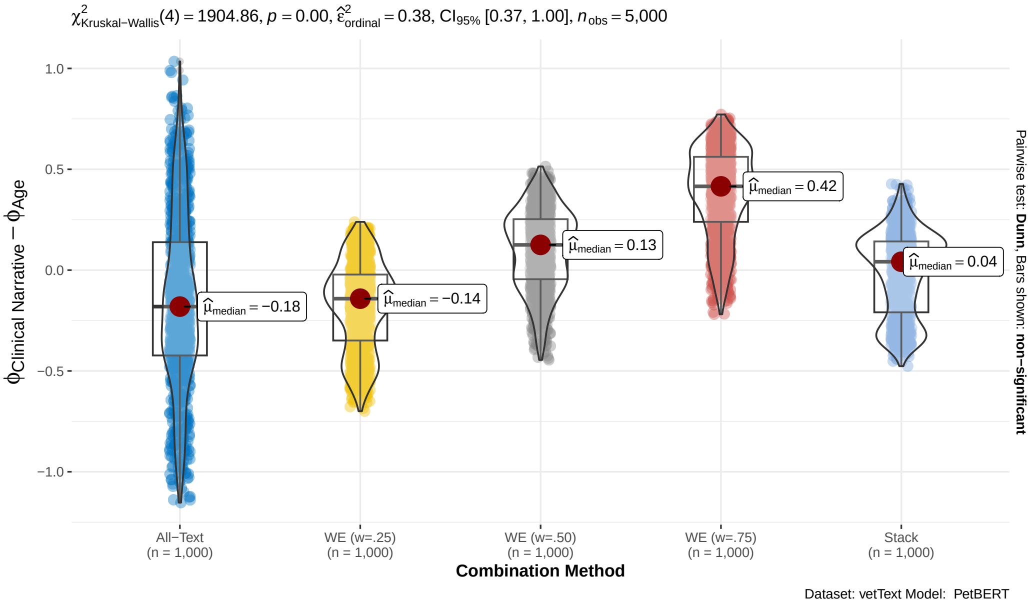
PetBERT, by method	All-Text	WE 25	WE 50	WE 75
WE 25	0.31 (0.19)			
WE 50	0.36 (0.21)	0.72 (0.15)		
WE 75	0.41 (0.25)	0.55 (0.19)	0.71 (0.19)	
Stack	0.33 (0.21)	0.66 (0.15)	0.66 (0.17)	0.55 (0.20)

6.4.3 Comparing the Two Most Influential Features

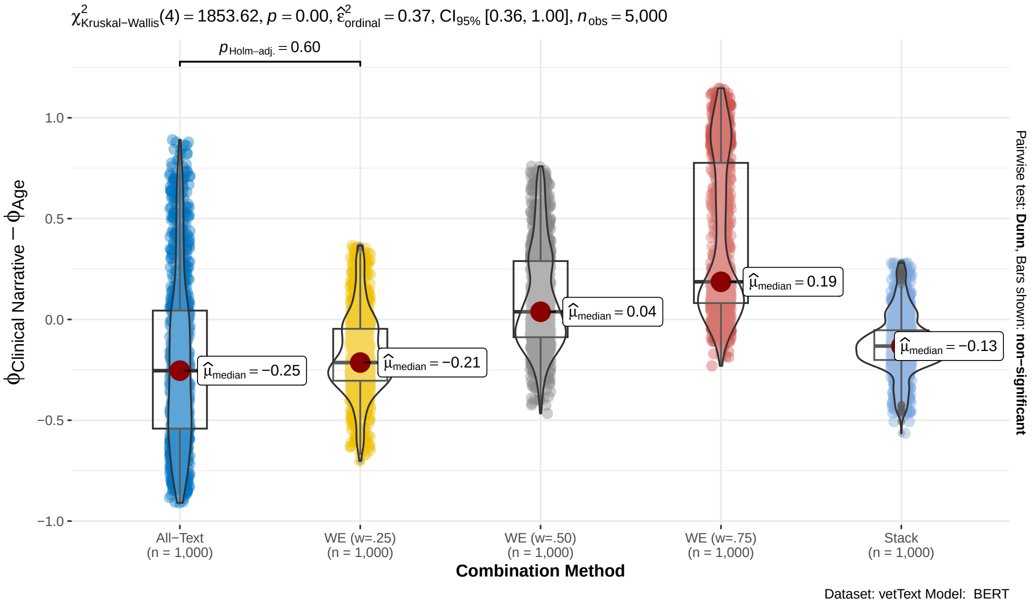
Here, we look at another way of comparing the different models; we aim to get a more general idea of how the two most influential features (*clinical narrative*, a text feature, and *age at consult*, a tabular feature) are treated using each of the five combination methods. For each of the 1000 instances, we examine the difference in feature importance between these two features. In Figure 6.3(a), we plot the difference between these two for each of the five combination methods when TM=PetBERT and repeat for TM=BERT in Figure 6.3(b). Once more, one can see greater importance being placed on *clinical narrative* than *age at consult* for PetBERT when compared to BERT, with all combination methods scoring a higher median difference. For *All-Text* experiments, one can see far longer tails in the difference distributions than the other methods. This again provides evidence of the importance of both features, differences further away from 0 indicating many cases where *age at consult* is key, *clinical narrative* is not, and vice versa. Furthermore, one can also see the increased reliance on text features, in this case, *clinical narrative*, with the difference growing more positive as w increases from 0.25 through to 0.75.

6.4.4 Top Phrases and Tabular Values

So far, we have considered all features as a whole, summing SHAP values for individual words to provide an overall score of importance for the entire feature, and comparing text features to tabular features. In Figure6.2 *clinical narrative* (a text feature) was seen to be the most influential. In a text-only context, one can use the original SHAP library to identify individual word pieces that are the most influential across an entire set of predictions. Using the multimodal SHAP, for the first time, one can directly compare individual words to individual tabular feature values. To avoid analysing fragments of words, we set a grouping threshold such that word pieces are grouped into words and phrases.



(a)



(b)

Figure 6.3: The difference in feature importance (SHAP) between the *clinical narrative* free-text field and *age*, plotted for PetBERT (a) and BERT (b), for each of the 1000 instances, for each of the five combination methods. The highly significant p-values (both p=0.00) for both Kruskal-Wallis tests indicate that the median differences for each of the combination methods are not the same; 38% and 37% of the variance in the differences are down to the changing of combination method for (a) and (b) respectively. Comparing (a) and (b), one can see higher median differences for PetBERT, indicating a higher reliance on the clinical narrative than BERT. As for combination methods, the preference for the text feature increases as w, the text model weighting, increases.

Using the optimal model, PetBERT *Weighted-Ensemble*, w=0.50, we look at the 1000 instances and find the phrases and tabular values that were the most influential. As a comparison, we also repeat the analysis for BERT *Weighted-Ensemble*, w=0.50. For those that appear more than once, we take a mean average. The top and bottom 20 items are found in Table 6.5, where top and bottom refer to those that contribute the most towards predictions of *alive* and *dead*, respectively. Looking more broadly at the top and bottom 100 phrases and tabular values, the tabular feature *age at consult* dominates. For the PetBERT model, of the top 100 entries, all 100 were based on low *age at consult* values and 89 entries from the bottom 100 were based on high *age at consult* values.

Similarly, in the BERT model, 99 out of the top 100 and 89 of the bottom 100 were from low and high *age at consult* values, respectively. In Figure 6.2, we identified that *age at consult* was an important feature, and here we indeed confirmed that older animals are more likely to be found within predictions of *dead* and vice verse for predictions of *alive*. The median age at consultation was 7.1 for the highest and 13.58 for the lowest. The median average for *age at consult* in the dataset was calculated to be 6.29.

Table 6.5: Phrases or instances of tabular features with the highest and lowest SHAP values across all instances for the best performing model, PetBERT *Weighted-Ensemble* w=0.50 and its BERT equivalent. A positive number indicates that the phrase contributes towards a prediction of *alive*, whereas a negative number contributes towards a prediction of *dead*.

(a) BERT *Weighted Ensemble*, w=0.50

value	phrase
0.25	age_at_consult = 0.61
0.25	alert and responsive hydration normal.
0.24	age_at_consult = 1.55
0.24	age_at_consult = 0.37
0.24	age_at_consult = 0.93
0.24	age_at_consult = 1.19
0.24	age_at_consult = 0.99
0.24	age_at_consult = 3.17
0.24	age_at_consult = 0.69
0.24	age_at_consult = 1.34
0.24	age_at_consult = 0.29
0.24	age_at_consult = 1.36
0.24	age_at_consult = 1.51
0.24	age_at_consult = 1.83
0.23	age_at_consult = 2.29
0.23	age_at_consult = 3.07
0.23	age_at_consult = 1.59
0.23	age_at_consult = 0.27
0.23	age_at_consult = 0.23
0.23	age_at_consult = 1.32
...	...
-0.14	age_at_consult = 18.01
-0.14	age_at_consult = 16.08
-0.14	age_at_consult = 16.25
-0.14	age_at_consult = 17.91
-0.14	and has been eating more the past couple of weeks.
-0.14	age_at_consult = 17.85
-0.14	age_at_consult = 20.9
-0.14	lost weight
-0.14	age_at_consult = 17.9
-0.14	age_at_consult = 19.24
-0.14	age_at_consult = 17.21
-0.14	thining
-0.14	/ less mobile.
-0.16	Strong pulses.
-0.16	generally slowing
-0.16	hearing with old age
-0.18	down with age
-0.20	QOL etc
-0.20	age related?
-0.21	some mucless mass loss but

(b) PetBERT *Weighted Ensemble*, w=0.50

value	phrase
0.25	age_at_consult = 0.61
0.24	age_at_consult = 1.55
0.24	age_at_consult = 0.37
0.24	age_at_consult = 0.93
0.24	age_at_consult = 1.19
0.24	age_at_consult = 0.99
0.24	age_at_consult = 3.17
0.24	age_at_consult = 0.69
0.24	age_at_consult = 1.34
0.24	age_at_consult = 0.29
0.24	age_at_consult = 1.36
0.24	age_at_consult = 1.51
0.24	age_at_consult = 1.83
0.23	age_at_consult = 2.29
0.23	age_at_consult = 3.07
0.23	age_at_consult = 1.59
0.23	age_at_consult = 0.27
0.23	age_at_consult = 0.23
0.23	age_at_consult = 1.32
0.23	age_at_consult = 0.24
...	...
-0.14	age_at_consult = 17.06
-0.14	age_at_consult = 18.22
-0.14	age_at_consult = 17.72
-0.14	age_at_consult = 17.57
-0.14	age_at_consult = 16.09
-0.14	age_at_consult = 18.01
-0.14	age_at_consult = 16.08
-0.14	age_at_consult = 16.25
-0.14	age_at_consult = 17.91
-0.14	age_at_consult = 17.85
-0.14	age_at_consult = 20.9
-0.14	age_at_consult = 17.9
-0.14	age_at_consult = 19.24
-0.14	age_at_consult = 17.21
-0.15	ongoing history
-0.15	for older cat
-0.19	of nasal tumour
-0.23	hearing with old age
-0.24	his age
-0.28	generally slowing down with age

Table 6.6: Word and phrases with highest and lowest feature importance (SHAP) values, grouped by high-level category. [High/Low] represents those that contribute most to a prediction of [alive/dead]. Phrases from BERT *Weighted-Ensemble*, *w*=0.50 and PetBERT *Weighted-Ensemble*, *w*=0.50 are found on the left and right respectively. PetBERT’s increase in performance can be attributed to its greater likelihood of identifying shorthand or medical terms, as demonstrated by the increased frequency and greater complexity of words in the ‘Medications’ and ‘Vaccinations’ categories. N.B. "DUDE" = ’Defecating, urinating, drinking and eating’, "BAR" = ’Bright And Responsive’, "NAD" = Nothing Abonroaml Detected, "CE" = ’Clinical Examination’, "f/w" = flea and wormer treatment, "kc" = kennel cough "nobivac tricat" = vaccine for feline calicivirus, feline herpes virus type 1 and feline panleucopenia virus, "rhd" = rabbit haemorrhagic disease vaccine, "lepto4" = Canine leptospirosis vaccine, "DHP" = distemper, hepatitis (canine adenovirus) and canine parvovirus vaccine, "BCS" = ’Body Condition Score’

Contributing most to a prediction of *alive*

BERT *Weighted-Ensemble*, *w*=0.50

Category	N	Example
Symptoms and Health Conditions	22	"hydration good", "moulting", "DUDE normal" "DUDE all ok"
Veterinary Treatments and Procedures	15	"express anal glands", "skin improving", "deep oral exam", "trimmed"
Physical Examination Findings	11	"ears are fine", "coat good", "nothing abnormal detected", "perineum normal"
Medications	8	"advise drops", "analgesia" "wormer", "spot on"
Advice Given to Pet Owners	7	"explained to owner", "Advised regular bathing", "bring back if concerned"
Vaccination	6	"administered vaccine", "vaccine", "booster given"
Pet Behaviour	6	"very well behaved", "biting", "not biting, "demean"

PetBERT *Weighted-Ensemble*, *w*=0.50

Category	N	Example
Symptoms and Health Conditions	18	"spay wound", "checks over fine", "No concerns", "BAR"
Medications	15	"worming", "endectrid", "easecto", "quantex"
Physical Examination Findings	14	"abdo palp NAD", "otherwise NAD", "CE unremarkable"
Advice Given to Pet Owners	13	"advised joint supplements", "adv re neuter", "discussed kc vaccine"
Vaccinations	12	"nobivac tricat", "rhd", "lepto4", "DHP/ L4 + KC given"
Weight and Body Condition score	6	"BCS 5/9 good growth.", "nice weight", "28kg", "8kg"
Dental Conditions and Treatments	5	"teeth are great", "Dentition" "teeth good", "teeth clean"

Contributing most to a prediction of *dead*

Category	N	Example
Symptoms and Health Conditions	25	"muscle mass loss", "dysuria", "constipated", "vomiting"
Age Related Issues	14	"age-related hearing loss", "old dog", "given age", "getting very old"
Food Diet	10	"not eating for 3 days", "drinking ok", "not eating well" "been eating more",
Owner’s Observations and Concerns	10	"O reports is drinking", "o aware decline inevitable", "o concerned coughing"
Medications	8	"Continue with steroid", "butorphanol", "prescription", "prednisolone"
Weight and Body Condition Score	8	"seems to have lost weight", "lost weight","lost 100g"
Vitals and Physical Examination Findings	7	"Bladder not palpable", "exam - senile", "Strong pulses" "R thyroid slightly enlarged",

Category	N	Example
Diagnoses	24	"mammary tumours", "Bladder cystitis", "osteoarthritis"
Symptoms and Health Conditions	14	"mobility issues", "Blind", "Weak", "ulcerated"
Quality of Life and Euthanasia Considerations	13	"euthanase", "palliative", "medical management", "quality of life"
Medical Procedures Further Testing	10	"bloods", "biopsy results", "Ultrasound", "drain"
Age Related Issues	8	" slowing down with age", "age related?", "old age", "surgery too risky with age"
Medications	6	"prednisolone", "mirtazapine", "chemo", "Vivitonin"
Owner’s Observations and Concerns	6	"o aware decline inevitable", "doesn’t want investigation", "o mentioned happier"

Continuing with our analysis of the top performing PetBERT model (*Weighted-Ensemble* w=0.50) and its BERT equivalent, we assess which types of phrases are most influential and how they compare to tabular features other than *age at consult*. We remove the dominant tabular feature and examine the new top 100 and bottom 100 lists of phrases and values. To summarise this information, we group each term into 10-12 high-order categories for each list. We count the items in each category and report the seven most populous groups for the top and bottom list, both for PertBERT and BERT. These results are displayed in Table 6.6, along with examples that typify each group. This representation conveys that the clinical language focused on by each model differs significantly, even if they fall under the same designated category.

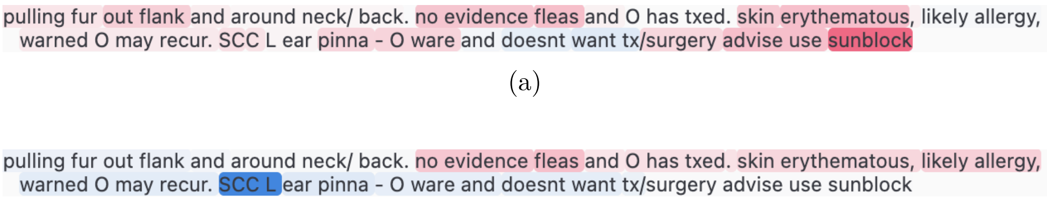
We further illustrate this point with a specific example shown in Figure 6.4, once more using *Weighted-Ensemble*, *w*=0.50. In this case, the true outcome was *alive*; the PetBERT *Weighted-Ensemble* predicted this correctly, whereas the BERT equivalent did not. One can see that both text models recognise 'no evidence fleas' as a positive sign. Similarly, erythematous - a typically non-serious reddening of the skin - was contributed towards an *alive* prediction for both models. However, the critical difference was that PetBERT identified 'SCC L' as shorthand for Squamous Cell Carcinoma, Left (ear) and a clear indicator that this particular animal is not likely to survive. The BERT model did not recognise this as the case and, in fact, regarded 'SCC L' as a positive sign for this animal.

6.5 Discussion

An abundance of data lies within the vast volumes of EHRs collected by initiatives such as SAVSNET. These records extend far beyond textual narratives alone, offering a diversity of modalities to be explored. Nevertheless, the path to harnessing the full potential of these rich datasets is challenging. While immensely powerful, the nature of deep learning frameworks becomes a source of complexity in the context of multimodality predictions. The principal challenge in this endeavour is the innate need for explainability within these frameworks, limiting the ability to extract comprehensive insights from complex AI models and their predictions. This is of paramount importance in the clinical domain, where transparency and interpretability are critical for gaining trust and acceptance among healthcare professionals and regulatory bodies.

This chapter continues and applies our prior research to this field, allowing us to get insight into a host of multimodal methods for the first time. We use the multimodal masking framework designed to engage in feature masking based on their respective modalities, ensuring uniform and consistent treatment of features, therefore fostering predictability in unimodal and multimodal contexts. This addresses the challenge of generating SHAP explanations for multimodal inputs, extending beyond the traditional unimodal context. In this study, we applied the framework to a text-tabular dataset of EHRs sourced from first-opinion veterinary practices across the United Kingdom to understand the features associated with mortality.

We examined the level of importance assigned to each feature and found a diverse set of preferences between combination methods, but overall, a strong preference for the free-text *clinical narrative* and the tabular feature *age at consult*. For all combination methods tested, PetBERT found an increased relative importance



pulling fur out flank and around neck/ back. no evidence fleas and O has txed. skin erythematous, likely allergy, warned O may recur. SCC L ear pinna - O ware and doesnt want tx/surgery advise use sunblock

(a)

pulling fur out flank and around neck/ back. no evidence fleas and O has txed. skin erythematous, likely allergy, warned O may recur. SCC L ear pinna - O ware and doesnt want tx/surgery advise use sunblock

(b)

Figure 6.4: Contrasting explanations for an example where BERT (a) was incorrect and PetBERT (b) was correct, both *Weitghed-Ensemble*, w=0.50. Words and phrases coloured [red/blue] indicate those that the model found to contribute towards a prediction of [alive/dead]. Both ensembles share an identical tabular model; therefore, we show the subset of the input from the clinical narrative to better exhibit the difference in explanations.

for *clinical narrative* when compared to BERT, a difference most pronounced in the *Stack-Ensembles*. The *Weighted-Ensembles* appeared similar to each other, with no other comparison between combination methods scoring higher than that between the *Weighted Emsembles* w=0.50 and w=0.25. This is consistent with what one would expect as the same model structure is used, differing by a linear transformation.

More generally, we found that changing the combination method had a greater impact on which features were attended to than changing the text model. For this particular dataset, both the underlying text and tabular models scored similarly well. Therefore, the differences in F1 scores for the ensemble models were also similar despite differing feature contributions. A much-reduced importance for other tabular features in the *All-Text* models suggests that information contained in these features, such as a cancer diagnosis in *neoplasms*, is already broadly covered in the free-text *clinical narrative* and is therefore ignored by the text models. However, in the same vein, we suggest that not all information was captured as results in Table 6.1 show that for both text models, *All-Text* was outperformed by *Weighted-Ensemble* w=0.50. The ICD labels represent a broad clinical history of a given animal; therefore, there will be instances where there is an overlap of events within the ICD set and the free-text narrative and other times where the label represents clinical events from many years prior.

Unsurprisingly, there was a notable enhancement in model performance arising from the additional pre-training of PetBERT on 500 million tokens from veterinary clinical narratives when compared to the standard BERT-base model. F1 and accuracy performance improvements of 2% were observed, compared to the BERT-base model employing the same evaluation strategy within the best-performing method. While this outcome aligns with our initial expectations, our methodological analysis offers insight into the divergent utilisation of distinct data modalities within the models.

To understand the performance of both models on a more granular level, we explored the types of words, phrases and tabular values that were most influential for each model. This was overwhelmingly predominated by *age at consult.* A general and expected trend emerged, suggesting older ages were more likely to die than lower ages. To better discern the difference between BERT and PetBERT, we looked at the words, phrases and tabular values without the presence of *age at consult.* Notably, there were overlaps observed here; for instance, discussions around vaccination were a common theme associated with animals predicted to be alive within the next 28 days. This emphasis between the two models aligns with the inherent logic that one typically would not vaccinate a severely ill animal.

Other examples include references to 'no concerns' categorised into the 'physical examination findings', which appeared as the third most common category of phrases in both PetBERT and BERT. Phrases such as 'other NAD [Nothing Abnormal Detected]' and 'CE [Clinical Examination] unremarkable' are unlikely to be used for animals expected to die imminently. Conversely, for words and phrases attributed to an animal approaching death, we observed a shared emphasis on discussions related to symptoms and health conditions. However, the significance of this indicator was more pronounced in the BERT model than in PetBERT. This approach also revealed that PetBERT exhibits a heightened 'understanding' across veterinary clinical free-text. This advantage enables PetBERT to interpret the veterinary clinical language associated with these subject matters more effectively than regular English, on which BERT was initially trained. Distinctly, PetBERT selected more definitive diagnoses as a more significant indicator, such as in 'mammary tumours'. Overall, words and phrases around cancers and mass growths emerged as noteworthy indicators in both models, although more so in PetBERT.

Although both models identified signs of vaccination as a positive indication, the words and phrases differed. PetBERT selected specific vaccination names such as 'lepto4' and 'nobivac tricat', whereas BERT used more generalised terms such as 'booster' and 'vaccine'. When thinking about the generalised corpora that BERT was trained on, there is a frequent theme where veterinary-specific terminology is not well understood, but phrases shared with human clinical medicine are present. Another example is within the ‘medications’ category, BERT’s utilisation of drug names ‘steroids’, ‘butorphanol’, and ‘prednisolone’ are all authorised drugs used frequently in human medicine. However, drugs such as ‘Vivitonin’, which was utilised by PetBERT, are authorised solely for dogs in the UK. Increased comprehension of phrases pertinent to diagnostic diseases, drug names, and diagnostic tests could attest to PetBERT’s superior clinical proficiency.

The EHR data used in this study offers valuable insights into death occurrences among the dogs and cats analysed. However, the depth of this analysis is contingent upon the information recorded by the veterinary practitioners. Consequently, these models are limited to capturing only those conditions or events explicitly documented in the EHRs. Any unrecorded or overlooked aspects cannot be accounted for in this analysis. The framework we have employed is fundamentally underpinned by SHAP and transformers, both of which are computationally expensive. This computational burden can lead to prolonged processing times, potentially limiting the scalability of this approach, especially when working with larger datasets or in real-time clinical settings.

In the context of *All-Text*, a single style of string template was the exclusive choice. In future investigations, exploring the impact of diverse template styles on explanations could be beneficial. The initial study developed a classifier that identified animals that have died with an F1 score exceeding 98.3%. Both the previous study and this study characterised the outputs for use within the prediction of mortality risk modelling. Therefore, it is likely that some data used within this study was incorrectly misclassified. While this level of misclassification is unlikely to impact the overall findings substantially, it is a point of consideration when interpreting individual predictions or decisions based on the model’s output. Furthermore, the dataset used in this study was sourced from participating veterinary practices. Consequently, the findings presented here may only partially represent the broader UK companion animal population. As the national coverage of participating practices within the Small Animal Veterinary Surveillance Network (SAVSNET) expands, these issues of coverage bias may be mitigated.

To conclude, this study investigated the complex dynamics governing the interaction between deep learning models and data modalities in the context of veterinary clinical EHRs. The findings suggest that the changing modality combination method has a more substantial influence on which features models find important, whereas both text models in this study tended to rank similar features as important. Additionally, PetBERT, having undergone additional pre-training, demonstrated enhanced comprehension of phrases related to cancer, drug names, and diagnostic tests, suggesting its superior proficiency in veterinary clinical language compared to BERT. The study highlights the capacity of language models to extract valuable insights from clinical narratives, providing contextual factors that inform predictions regarding animal well-being. The comparative analysis of both modalities within a uniform framework has significantly enabled the comprehension and interpretation of the overall model prediction and enabled a per-input feature comparison, regardless of whether that be a text or tabular value.

Chapter 7

Concluding Remarks

Each chapter in this thesis has seen the introduction of a novel technique, dataset or application, first to use language models to enhance the explainability of numerical outputs and secondly to address the research gap for text-tabular explanations. This chapter will summarise our contributions and discuss limitations and future work.

7.1 On the Importance of Continued Explainability Research

The landscape of AI has witnessed a significant evolution, as detailed in Chapter 2 of this thesis, particularly through the emergence and rapid development of LLMs. These advancements have not only pushed the boundaries of what AI can achieve but also have catapulted themselves into the forefront of public consciousness, further propagating this cycle of advancement. Amidst this rapid growth, I argue that it is imperative that explainability keeps pace with the broader field and that we continue to find ways of making the output of machine learning decisions more approachable and interpretable.

In response to this need, in this thesis I aimed to extend the horizons of explainability just that little bit further by harnessing language models expanding explainability to the text-tabular subfield, therefore bringing explanations to a wider audience in an accessible and understandable manner.

This began in Chapter 3, where the focus was on leveraging the capabilities of LLMs to translate technical performance metrics into easily comprehensible textual descriptions. This initiative underscored the potential of language models in making complex information more accessible and understandable to a broader audience.

Chapter 4 built upon these foundations, redefining what a machine learning explanation could look like by generating textual narratives rather than numerical or graphical outputs. This chapter not only defined another new data-to-text task but also directly engaged with explaining machine learning outputs, providing an additional layer of context to help remove the barriers to comprehension.

Following the exploration of explanations as textual narratives, I ventured deeper into the realm of explainability in Chapter 5. Acknowledging a research gap highlighted in Chapter 2, concerning the lack of explainability methods for text-tabular datasets, I created a novel multimodal masker for SHAP to bridge this gap. This adaptation was significant as it marked the first time it was possible to bring one of the most widely used explainability methods to the text-tabular context, marking a significant leap forward in multi-modal interpretability. Through a series of experiments, this chapter scrutinised the effects of porting the previous unimodal masker for use in multimodal scenarios, unveiling insights into feature importance distribution and the efficacy of different explanation combination methods and text models.

Using the multi-modal masker, in Chapter 6 I showcased the practical implications and insights that can be gleaned from extending explainability to the text-tabular domain. Through a detailed analysis of the SARS veterinary dataset, this chapter shed light on how the tailored PetBERT was able to outperform BERT-base, notably highlighting the superior capability of PetBERT in understanding veterinary shorthand. This deep dive demonstrated the importance of tailored pre-training for specific domain applications, but more significantly emphasised the tangible benefits that can be derived from continuing to advance the field.

To conclude, I hope that this thesis underlines the value and the importance of continued progression within the explainability domain and that rapid advancements across the wider field are a motivation and not a distraction in making models more transparent, understandable, and ultimately, more trustworthy.

7.2 Strategies for Data-to-Text Applications

In this thesis, the central focus has been on making the outputs of machine learning more comprehensible to a broader audience. The work undertaken in Chapters 3 and 4 was pivoted around using of LLMs to translate what is conventionally represented in numbers or graphs into natural language. In both chapters, the complexity of the tasks warranted an in-depth understanding of the subject field. Therefore I solicited the help of computer science experts to curate high-quality datasets for both of these new and specialised data-to-text tasks, a fundamental component of these chapter's contribution. To make the most of a relatively small number of samples, I used a number of techniques and strategies that I believe would be useful for other data-to-text tasks, especially those also with limited data availability.

One approach successfully deployed in Chapters 3 and 4 was data augmentation. Data augmentation, often tricky to do accurately with text tasks, was possible in this scenario. When feeding in metrics as input in Chapter 3, I increased the size of the dataset with no additional text narrations simply by randomising the order in which the metrics were passed into the model. In this manner the model could learn not to focus on the ordering of the metrics, but instead the relationships between the metric names, values and the textual output.

Chapter 4 once more showed the efficacy of using data augmentation in a data-to-text task. When the numerical explanations were fit to a string template and their feature names were anonymised and replaced with a template, I recognised that this was a space for augmentation. Once more, this time by re-randomising the attribution of placeholders, I was able to cement the relationship between the input and the output and demonstrate the potential of data augmentation as an effective strategy in data-constrained data-to-text tasks.

Moreover, each chapter delves into additional methodologies for ingraining the necessary relationships within the models. In Chapter 3, I introduced the Metric Processing Unit as a novel means for the T5 and BART language models to assimilate the correlation between metric values and the generated textual output. This adaptation yielded the best performance across both model variants. Chapter 4 presented a greater challenge, given the expectation for the model to grasp the content of numerical explanations—which might be entirely or almost entirely unfamiliar. I found success when being more precise with what I was asking the model, reducing complexity and stripping back the problem to a question-answer task such that the models only have to provide targeted answers as opposed to a more generalised summarisation task. Here the T5 model achieved an overall accuracy of 91%.

7.3 Limitations and Future Work

While I believe this thesis offers substantial insight and advancements in the field, I acknowledge that there is room for further development. Although discussions of potential future work are mentioned in individual chapters, I consolidate and examine these aspects in detail here within the broader context of this work.

One primary limitation of the Natural Language Explanation tasks in Chapters 3 and 4 is the size of the datasets. Due to the availability and cost associated with acquiring assistance from computer science PhD candidates for data collection, the dataset is not as expansive as I would have preferred. Although employing experts was a deliberate choice to facilitate a dataset of high quality, the relatively small size could affect the generalisability and robustness of models trained on these datasets.

An interesting option for future work - particularly applicable to the Metric to Text task of Chapter 3 - would be to attempt to imbue a deeper sense of mathematical understanding into the models in order to learn the numerical relationships between each of the metrics. One possible approach to achieve this could be to incorporate this into a mathematical pre-training regimen or, alternatively, in an auxiliary parallel task during fine-tuning. This would allow models first to understand mathematical structures, properties, and relationships from a dedicated mathematical dataset. However, a risk is that one could be substituting mathematical understanding for reduced linguistic proficiency.

Furthermore, for these two tasks I confront one of the other principal challenges: precisely evaluating the quality of the generated tasks. Current evaluation metrics, primarily designed for translation and summarisation, do not always adequately capture the subtlety needed in this task. In particular, error analysis in Chapter 4 revealed a propensity of the models to make incorrect factual statements about a group of features; a comment summarising ‘all remaining features’, for example, needs to be exact and specific wording makes a difference. With the advent of advanced models such as GPT-4, in a future analysis, I would like to use these LLMs directly to develop a more effective method for evaluating the accuracy of statements.

In the aforementioned error analysis, I began to categorise sentences into different classes. If I were to label each sentence of the dataset with an indicator as to which type of statement is being made, I believe this would give the model yet more clues as to how the input relates to the output. Further motivation to do this comes from the question and answer section, where I demonstrated the value of breaking up the task into constituent parts.

In Chapter 5, the creation of the multi-modal masker opens the doors to a whole suite of analysis. I experimented with changing text models, combination methods, and datasets, but there are many more directions that further analysis could take. For example, I would like to extend this experimentation to examine the effects of using different styles of text templates. I would look for performance differences, but the principal would be to see whether this affects the features a model finds important. Moreover, this could shed light on which templates exacerbate the issues related to the unimodal masker.

Additionally, crafting model-specific implementations of the masker that accelerate the generation of explanations would be worthwhile. The original SHAP library now offers a whole host of explainers tailored to diverse model types, such as neural networks, GPU implementations, and tree-based models. Creating faster, more focused explanations could greatly enhance the method's utility and favour its adoption by a broader audience.

Another intriguing area for future work involves utilising the masker to track how explanations evolve across modalities during the model training process. Stopping training and generating explanations at various stages of training could reveal insights into which types of features are learned first and whether this differs across various combination methods and text models.

In Chapter 6, I applied the multimodal masker to do a deep dive on SAVSNET. This is a real-world dataset, so in future analysis, I would like to use the work I have done to inform future decisions. Although, more often than not, the models were able to handle veterinary shorthand, explicitly defining the shorthand present could be a method to improve model performance. A pet's age is the most important feature in this analysis, so much so that in the original PetBERT paper (Farrell et al., 2023b), separate models were trained for each of the three age bands, which had notably different performances. Extending our explainability analysis to models banded by age would give an insight as to which words and features cause a particular prediction for young, adult and old animals. In a similar fashion, Farrell et al. (2023b) also trained models to predict further than one month in the future -

another route for further analysis.

7.3.1 Epilogue

In the course of this thesis, I have endeavoured to present additional insights and contexts that underpin various decisions and tasks. Specifically, in Chapters 3 and 4, I took this on in the form of data-to-text generation, training models capable of crafting narratives that offer users an alternative textual perspective on the data being presented. Both of these tasks involved carving out a new niche in data-to-text, and in both instances, a new task and dataset were proposed and created. Although relatively small, they are high in quality, and I hope these can provide a springboard for future research.

As artificial intelligence becomes increasingly integrated into our daily lives, the demand for greater transparency and understanding of these systems intensifies. The forthcoming EU AI Act is set to establish stricter standards for the operation of black-box models, mandating adherence to principles of safety and fairness. In Chapter 5, I tackled a substantial research gap to enable users to generate explanations in a domain that previously was just not possible. It is our aspiration that this work will pave the way for researchers to gain a new perspective on why text-tabular models are making their decisions and advance the field of AI interpretability.

Chapter 8

Appendix

Table 8.1: Comparing the similarity of combination method feature rankings. Kendall's Tau is measured for each combination method with each other combination method, for each of the four text models, for each of the nine datasets. AT refers to *All-Text*, AT (U) refers to *All-Text (Unimodal)* and WE refers to *Weighted-Ensemble*.

(a) Airbnb

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.37					0.25					0.22					0.29	
WE .25	0.25	0.41				0.21	0.34				0.22	0.49				0.19	0.42
WE .50	0.24	0.43	0.80			0.23	0.40	0.79			0.20	0.45	0.76			0.20	0.47
WE .75	0.20	0.38	0.58	0.73		0.22	0.37	0.58	0.74		0.15	0.35	0.50	0.68		0.19	0.45
Stack	0.23	0.41	0.75	0.74	0.60	0.22	0.36	0.73	0.75	0.64	0.21	0.44	0.72	0.73	0.55	0.20	0.45

(b) Channel

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.26					0.10					0.23					0.29	
WE .25	-0.13	-0.17				0.01	-0.13				-0.04	-0.23				-0.06	-0
WE .50	-0.04	-0.08	0.87			-0.07	-0.03	0.87			0.05	-0.15	0.87			0.02	-0
WE .75	-0.03	-0.06	0.81	0.93		-0.11	0.03	0.78	0.90		0.09	-0.12	0.78	0.91		0.04	-0
Stack	-0.04	-0.06	0.64	0.68	0.67	0.00	-0.09	0.67	0.64	0.59	-0.01	-0.21	0.66	0.64	0.60	0.00	-0

(c) Fake

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.72					0.65					0.66					0.56	
WE .25	0.71	0.76				0.53	0.58				0.59	0.77				0.47	0.48
WE .50	0.76	0.79	0.90			0.57	0.63	0.95			0.62	0.79	0.96			0.48	0.40
WE .75	0.71	0.68	0.69	0.79		0.59	0.64	0.86	0.91		0.62	0.82	0.93	0.96		0.49	0.41
Stack	0.45	0.52	0.60	0.53	0.33	0.30	0.30	0.65	0.63	0.57	0.54	0.57	0.64	0.62	0.61	0.33	0.33

(d) Imdb

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.27					0.33					0.37					0.34	
WE .25	0.15	0.02				0.04	0.04				-0.17	-0.03				0.06	0
WE .50	0.29	0.20	0.81			0.16	0.23	0.80			-0.04	0.12	0.81			0.17	0
WE .75	0.32	0.28	0.65	0.82		0.21	0.29	0.62	0.80		0.08	0.26	0.58	0.74		0.22	0
Stack	0.05	-0.12	0.41	0.31	0.23	-0.13	-0.19	0.36	0.21	0.10	-0.40	-0.34	0.42	0.29	0.13	0.02	-0

(e) Jigsaw

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.06					0.05					0.01					0.10	
WE .25	-0.04	0.16				-0.04	-0.01				0.07	-0.15				0.08	0.0
WE .50	-0.04	0.17	0.95			-0.03	0.01	0.97			0.07	-0.14	0.96			0.07	0.0
WE .75	-0.03	0.17	0.91	0.93		-0.02	0.01	0.92	0.94		0.09	-0.14	0.90	0.93		0.08	0.1
Stack	-0.01	0.12	0.85	0.85	0.86	-0.04	0.01	0.82	0.83	0.84	0.08	-0.15	0.83	0.83	0.83	0.06	0.0

(f) Kick

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.50					0.35					0.50					0.43	
WE .25	0.40	0.56				0.26	0.53				0.35	0.53				0.36	0.59
WE .50	0.45	0.61	0.78			0.31	0.60	0.79			0.48	0.60	0.77			0.40	0.61
WE .75	0.44	0.57	0.57	0.78		0.31	0.56	0.55	0.75		0.51	0.59	0.56	0.78		0.41	0.55
Stack	0.37	0.54	0.55	0.60	0.60	0.26	0.56	0.61	0.70	0.61	0.42	0.58	0.67	0.75	0.66	0.40	0.58

(g) Prod

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.62					0.46					0.12					0.30	
WE .25	0.62	1.00				0.46	1.00				0.12	1.00				0.30	1.0
WE .50	-0.16	-0.06	-0.06			0.42	0.44	0.44			0.32	0.52	0.52			0.26	0.7
WE .75	-0.56	-0.94	-0.94	0.12		0.52	0.02	0.02	0.58		0.44	0.00	0.00	0.48		0.14	-0.3
Stack	0.62	1.00	1.00	-0.06	-0.94	0.46	1.00	1.00	0.44	0.02	0.12	1.00	1.00	0.52	0.00	0.30	1.0

(h) Salary

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.58					0.66					0.58					0.66	
WE .25	0.53	0.59				0.49	0.51				0.60	0.45				0.59	0.52
WE .50	0.56	0.55	0.93			0.54	0.56	0.89			0.61	0.39	0.93			0.63	0.54
WE .75	0.57	0.54	0.91	0.99		0.55	0.55	0.84	0.95		0.60	0.36	0.90	0.98		0.64	0.56
Stack	0.39	0.30	0.54	0.61	0.61	0.44	0.49	0.55	0.66	0.70	0.51	0.27	0.61	0.67	0.68	0.51	0.47

(i) Wine

	DistilBERT					BERT					DistilRoBERTa						
	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT	WE .25	WE .50	WE .75	AT (U)	AT
AT	0.88					0.86					0.83					0.68	
WE .25	0.28	0.27				0.35	0.34				0.42	0.42				0.38	0.37
WE .50	0.68	0.68	0.59			0.74	0.73	0.60			0.79	0.78	0.62			0.56	0.69
WE .75	0.82	0.85	0.41	0.83		0.83	0.81	0.48	0.89		0.82	0.83	0.46	0.84		0.63	0.77
Stack	0.70	0.66	0.45	0.79	0.75	0.74	0.72	0.43	0.78	0.79	0.65	0.63	0.55	0.73	0.64	0.52	0.59

Table 8.2: Comparing the similarity of text model feature rankings. Kendall's Tau is measured for each text model with each other text model, for each of the six combination methods, for each of the nine datasets. WE refers to *Weighted-Ensemble*, Dis.B refers to DistilBERT, Dis.R refers to DistilRoBERTa and DeB refers to DeBERTa.

(a) Airbnb

	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	0.69			0.69			0.91			0.91			0.89			0.77		
Dis.R	0.56	0.50		0.56	0.50		0.89	0.89		0.85	0.86		0.80	0.79		0.76	0.78	
DeB.	0.53	0.51	0.58	0.53	0.51	0.58	0.90	0.92	0.89	0.89	0.91	0.84	0.89	0.90	0.79	0.76	0.76	0.77

(b) Channel

	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	NaN			NaN			0.93			0.94			0.97			0.65		
Dis.R	NaN	NaN		NaN	NaN		0.94	0.95		0.95	0.95		0.97	0.96		0.63	0.65	
DeB.	NaN	NaN	NaN	NaN	NaN	NaN	0.94	0.93	0.95	0.94	0.93	0.95	0.96	0.94	0.96	0.66	0.65	0.65

(c) Fake

	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	0.76			0.76			0.90			0.85			0.71			0.51		
Dis.R	0.83	0.71		0.83	0.71		0.92	0.87		0.88	0.86		0.72	0.85		0.63	0.36	
DeB.	0.78	0.62	0.72	0.78	0.62	0.72	0.64	0.60	0.62	0.56	0.46	0.51	0.76	0.53	0.54	0.49	0.28	0.39

(d) Imdb

	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	0.44			0.44			0.9			0.9			0.91			0.65		
Dis.R	0.49	0.38		0.49	0.38		0.9	0.82		0.87	0.81		0.85	0.82		0.74	0.78	
DeB.	0.38	0.46	0.38	0.38	0.46	0.38	0.87	0.8	0.9	0.85	0.8	0.86	0.83	0.83	0.86	0.66	0.71	0.71

(e) Jigsaw

	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	0.34			0.34			0.95			0.93			0.91			0.83		
Dis.R	0.19	0.26		0.19	0.26		0.94	0.97		0.93	0.95		0.9	0.92		0.81	0.85	
DeB.	0.26	0.27	0.07	0.26	0.27	0.07	0.97	0.95	0.95	0.97	0.94	0.94	0.94	0.92	0.92	0.8	0.82	0.78

(f) Kick

	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	0.77			0.77			0.77			0.72			0.74			0.68		
Dis.R	0.62	0.63		0.62	0.63		0.81	0.74		0.77	0.70		0.79	0.76		0.64	0.62	
DeB.	0.67	0.67	0.65	0.67	0.67	0.65	0.80	0.72	0.83	0.76	0.67	0.81	0.79	0.74	0.84	0.64	0.63	0.67

(g) Prod

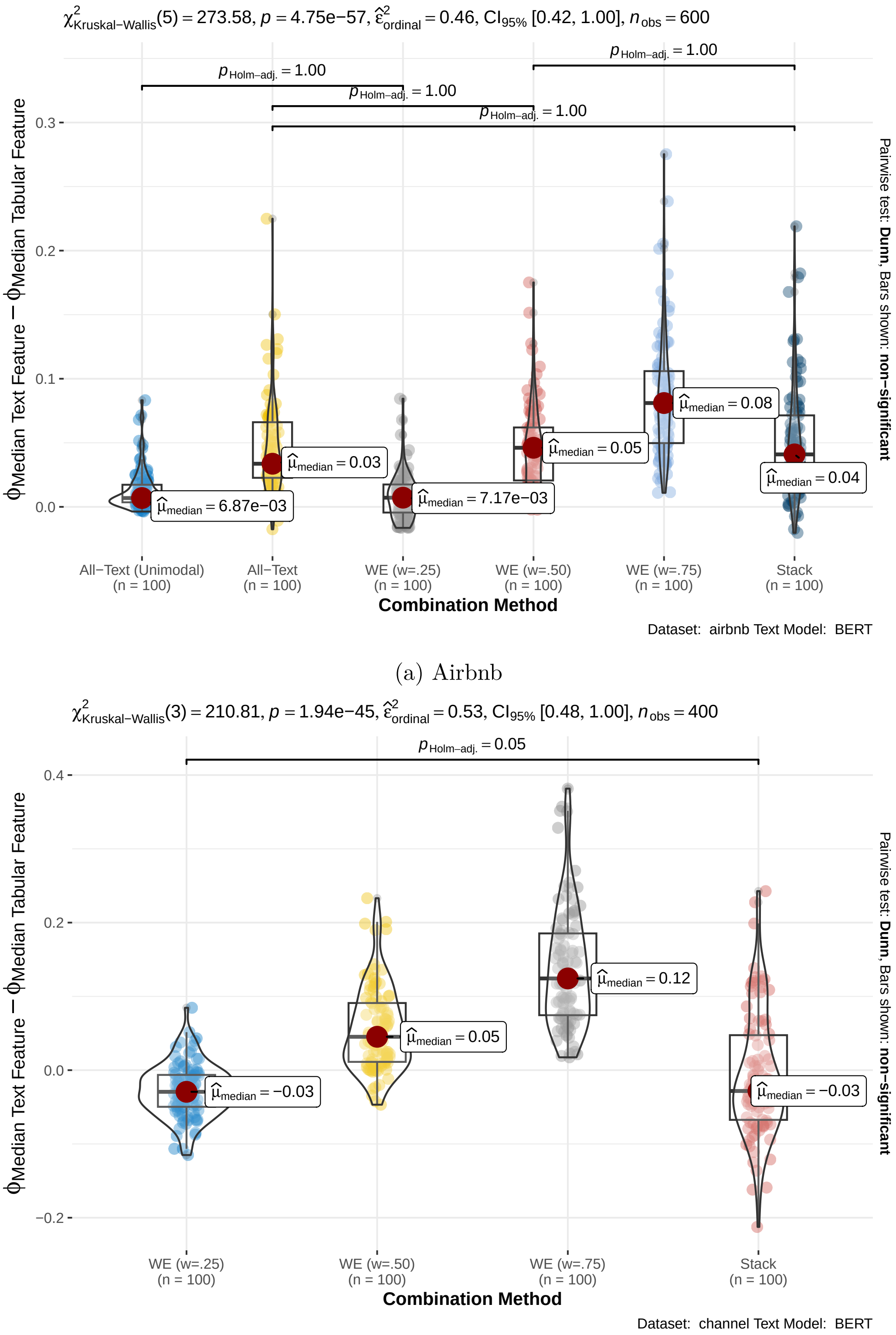
	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	1.00			1.00			1.00			-0.18			-0.08			NaN		
Dis.R	1.00	1.00		1.00	1.00		1.00	1.00		-0.26	0.72		-0.02	0.78		NaN	1.00	
DeB.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	-0.10	0.56	0.56	0.26	0.30	0.36	NaN	1.00	1.00

(h) Salary

	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	0.46			0.46			0.51			0.58			0.63			NaN		
Dis.R	0.42	0.35		0.42	0.35		0.49	0.6		0.53	0.66		0.55	0.71		NaN	NaN	
DeB.	0.53	0.57	0.43	0.53	0.57	0.43	0.36	0.54	0.51	0.45	0.6	0.56	0.48	0.66	0.62	NaN	NaN	NaN

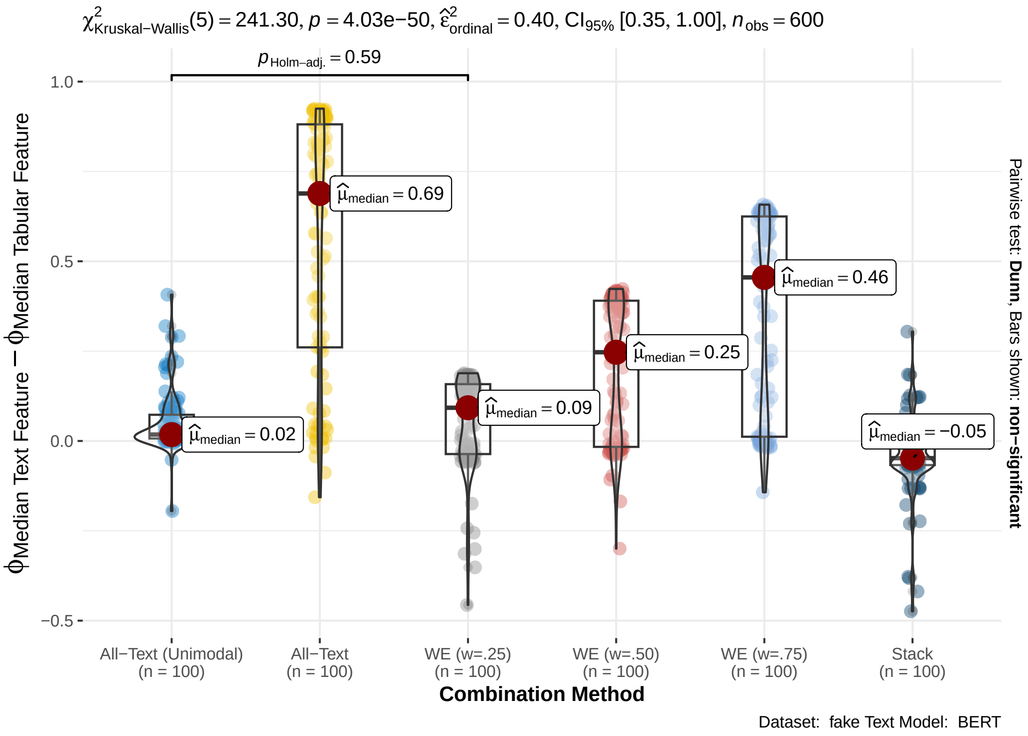
(i) Wine

	All Text			All Text (Uni.)			WE w=.25			WE w=.50			WE w=.75			Stack		
	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R	Dis.B	BERT	Dis.R
BERT	0.90			0.90			0.83			0.85			0.93			NaN		
Dis.R	0.92	0.89		0.92	0.89		0.80	0.80		0.85	0.88		0.88	0.85		NaN	NaN	
DeB.	0.82	0.83	0.85	0.82	0.83	0.85	0.74	0.76	0.75	0.74	0.80	0.80	0.83	0.82	0.84	NaN	NaN	NaN

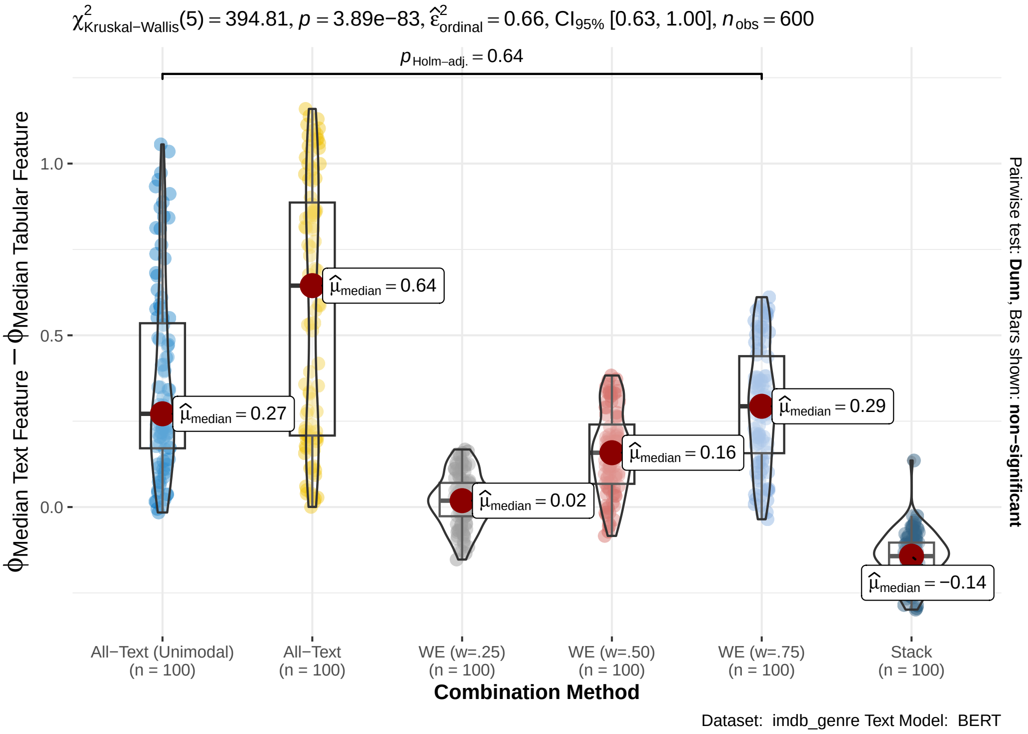


(a) Airbnb

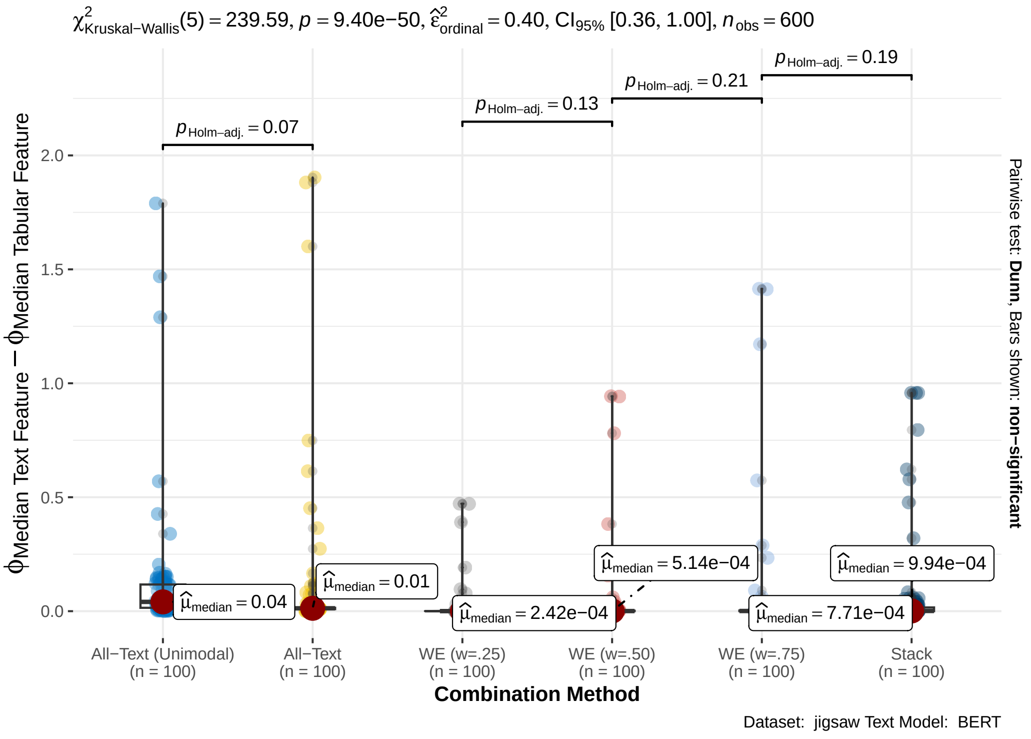
(b) Channel


$\chi^2_{\text{Kruskal-Wallis}}(5) = 241.30, p = 4.03\text{e}{-50}, \hat{\varepsilon}^2_{\text{ordinal}} = 0.40, \text{CI}_{95\%}$ [0.35, 1.00], $n_{\text{obs}} = 600$
$p_{\text{Holm-adj.}} = 0.59$
$\hat{\mu}_{\text{median}} = 0.02$
$\hat{\mu}_{\text{median}} = 0.69$
$\hat{\mu}_{\text{median}} = 0.09$
$\hat{\mu}_{\text{median}} = 0.25$
$\hat{\mu}_{\text{median}} = 0.46$
$\hat{\mu}_{\text{median}} = -0.05$
1.0
0.5
0.0
−0.5
ϕMedian Text Feature − ϕMedian Tabular Feature
All−Text (Unimodal) (n = 100)
All−Text (n = 100)
WE (w=.25) (n = 100)
WE (w=.50) (n = 100)
WE (w=.75) (n = 100)
Stack (n = 100)
Combination Method
Pairwise test: Dunn, Bars shown: non−significant
Dataset: fake Text Model: BERT

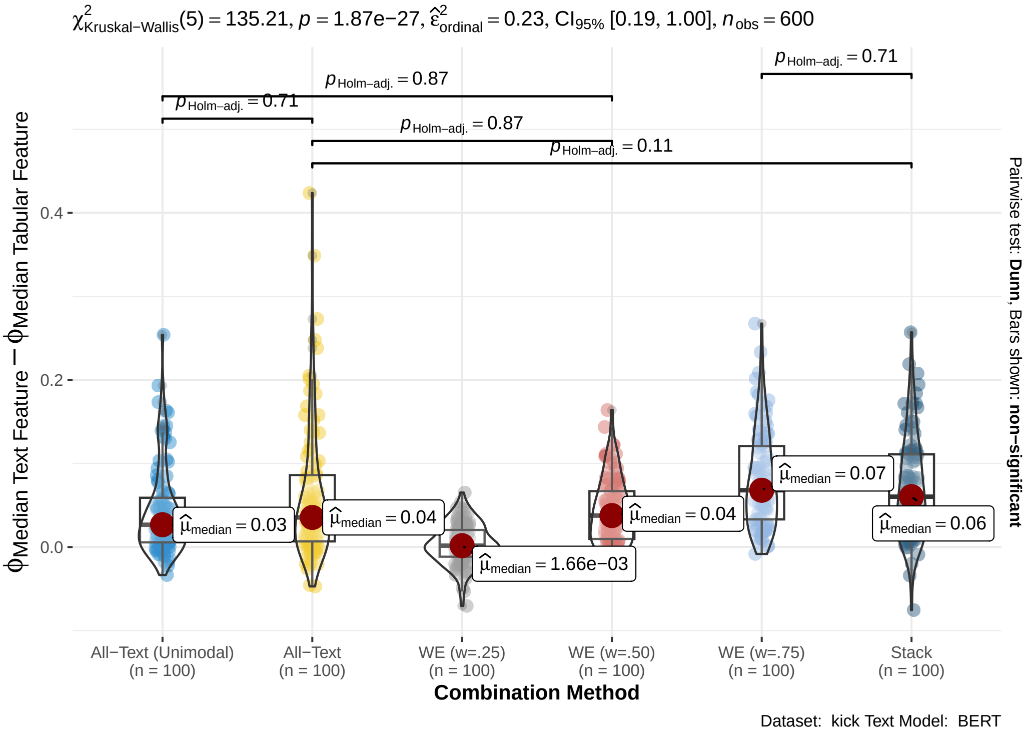
(c) Fake


$\chi^2_{\text{Kruskal-Wallis}}(5) = 394.81, p = 3.89\text{e}{-83}, \hat{\varepsilon}^2_{\text{ordinal}} = 0.66, \text{CI}_{95\%}$ [0.63, 1.00], $n_{\text{obs}} = 600$
$p_{\text{Holm-adj.}} = 0.64$
$\hat{\mu}_{\text{median}} = 0.27$
$\hat{\mu}_{\text{median}} = 0.64$
$\hat{\mu}_{\text{median}} = 0.02$
$\hat{\mu}_{\text{median}} = 0.16$
$\hat{\mu}_{\text{median}} = 0.29$
$\hat{\mu}_{\text{median}} = -0.14$
1.0
0.5
0.0
ϕMedian Text Feature − ϕMedian Tabular Feature
All−Text (Unimodal) (n = 100)
All−Text (n = 100)
WE (w=.25) (n = 100)
WE (w=.50) (n = 100)
WE (w=.75) (n = 100)
Stack (n = 100)
Combination Method
Pairwise test: Dunn, Bars shown: non−significant
Dataset: imdb_genre Text Model: BERT

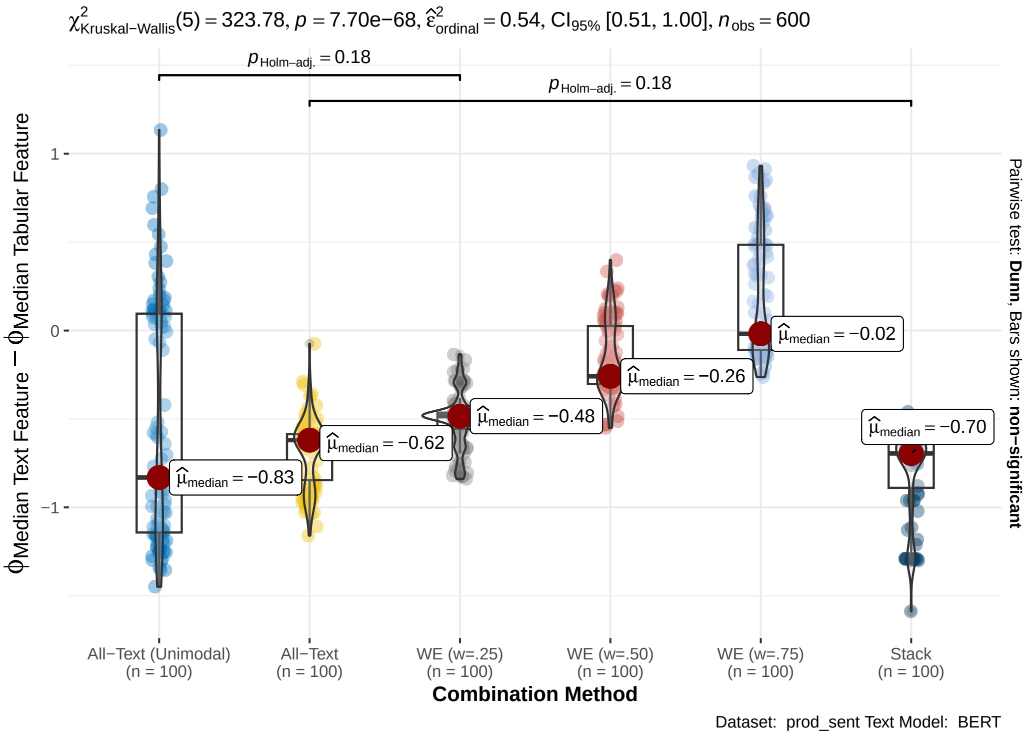
(d) Imdb



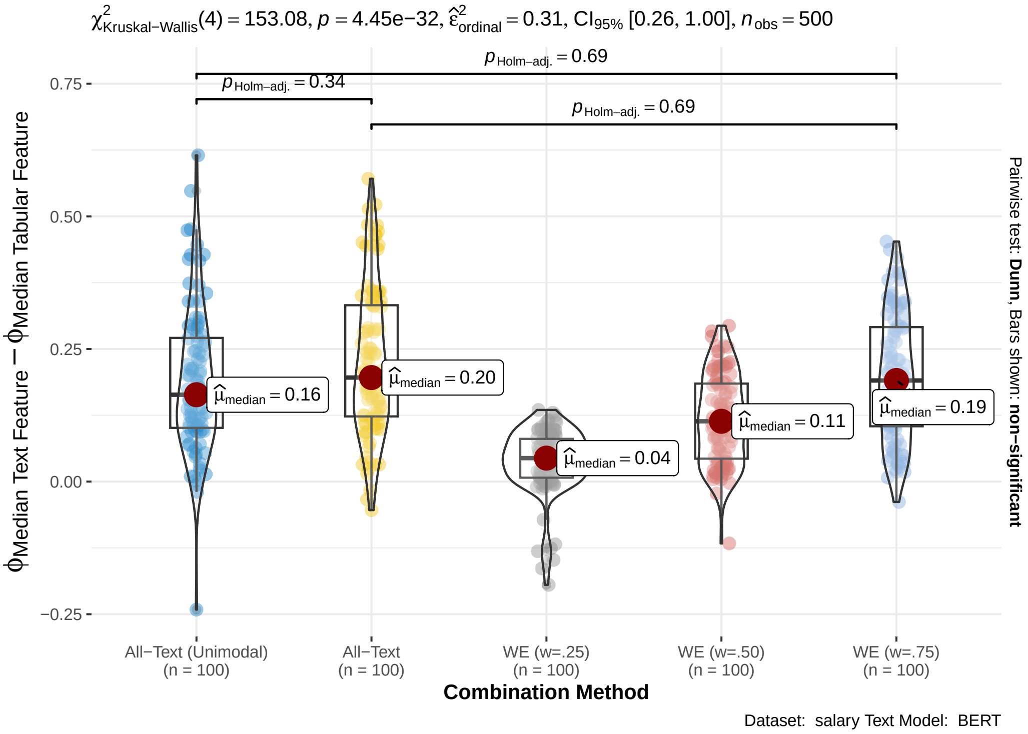
(e) Jigsaw



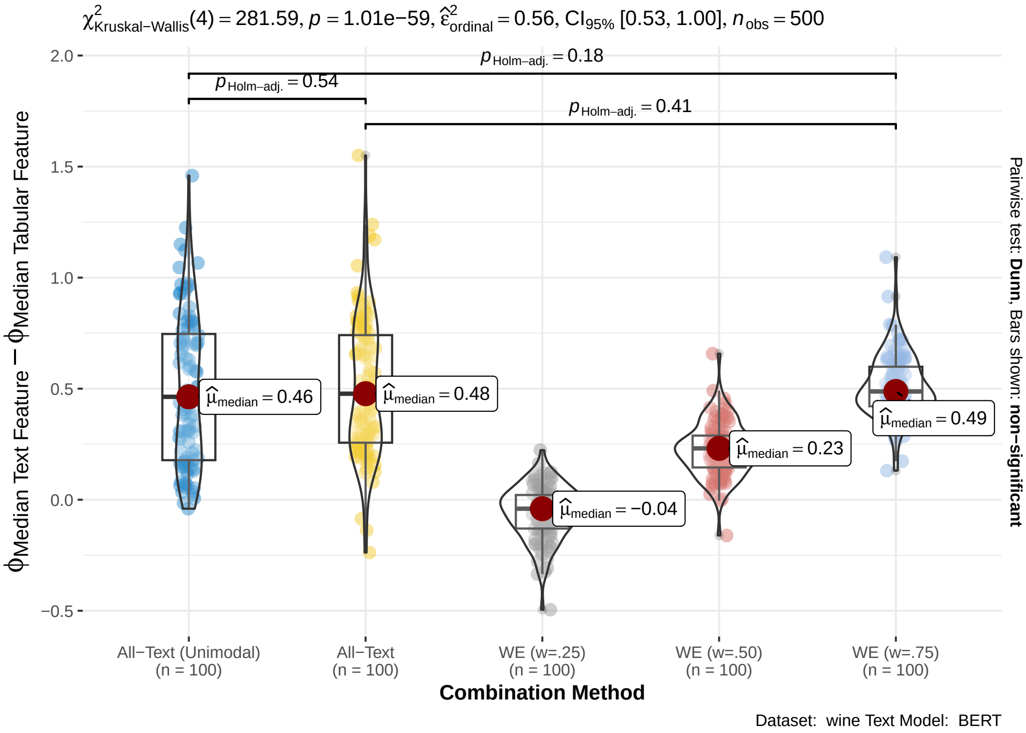
(f) Kick



(g) Prod

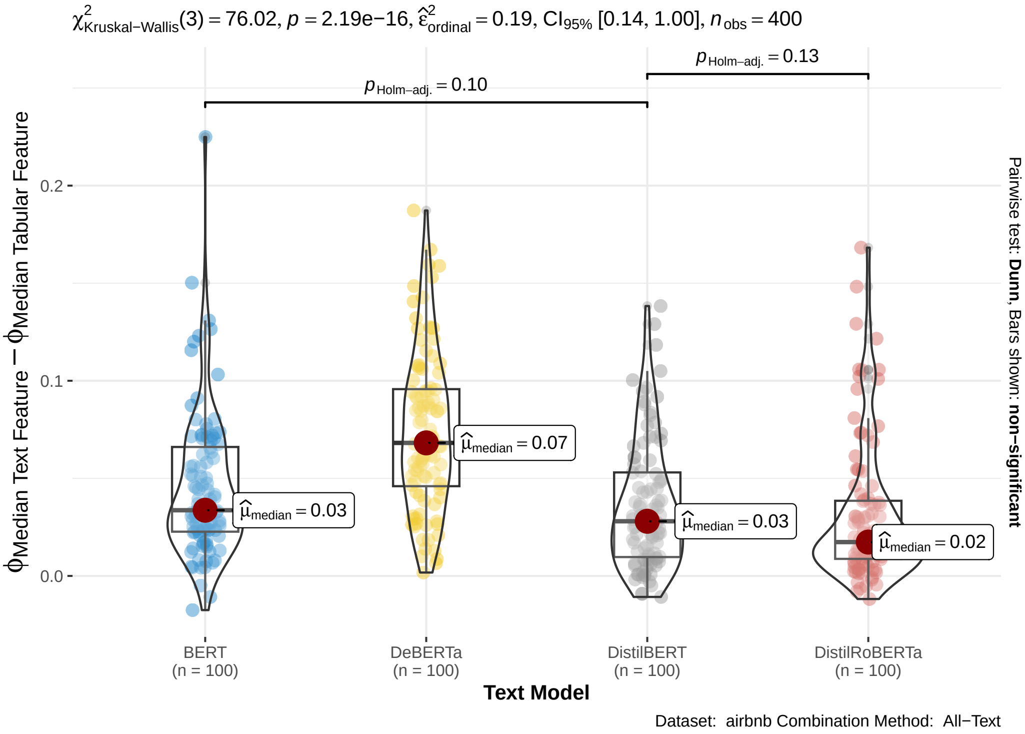


(h) Salary



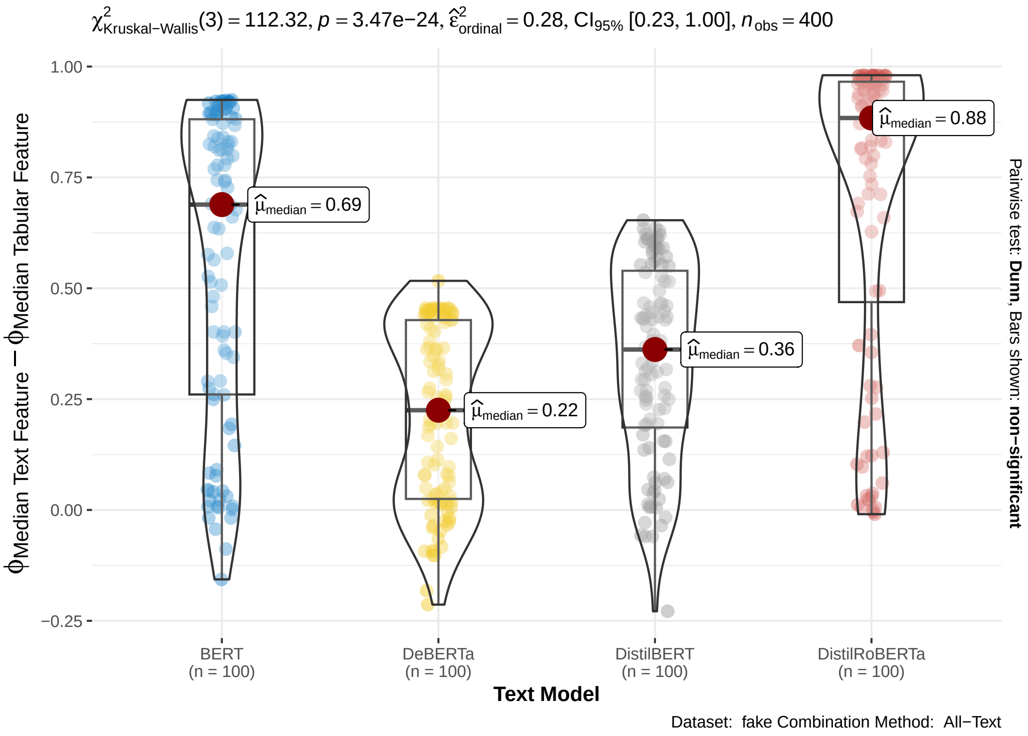
(i) Wine

Figure 8.1: In order to compare which combination method assigns more relative importance to text or tabular features, I plot $\Delta = \phi_{\text{Median Text Feature}} - \phi_{\text{Median Tabular Feature}}$ on the y-axis and combination method on the x-axis for each of the nine datasets with text model=BERT. Each differently coloured plot represents the distribution of Δ's for a single combination method, each with 100 coloured dots representing Δ for each of the 100 instances. For each plot, the distribution is also represented by a violin plot, which emphasises the non-normal distributions, a box plot, which indicates the spread of the data, and a labelled red dot, which indicates the median. A higher Δ indicates that a higher relative importance is assigned to text features over tabular features. The Kruskal-Wallis test statistic is significant for all nine of the datasets, meaning that the alternative hypothesis of not all medians being equal is accepted. WE refers to *Weighted-Ensemble* and w indicates the weighting of the text model.

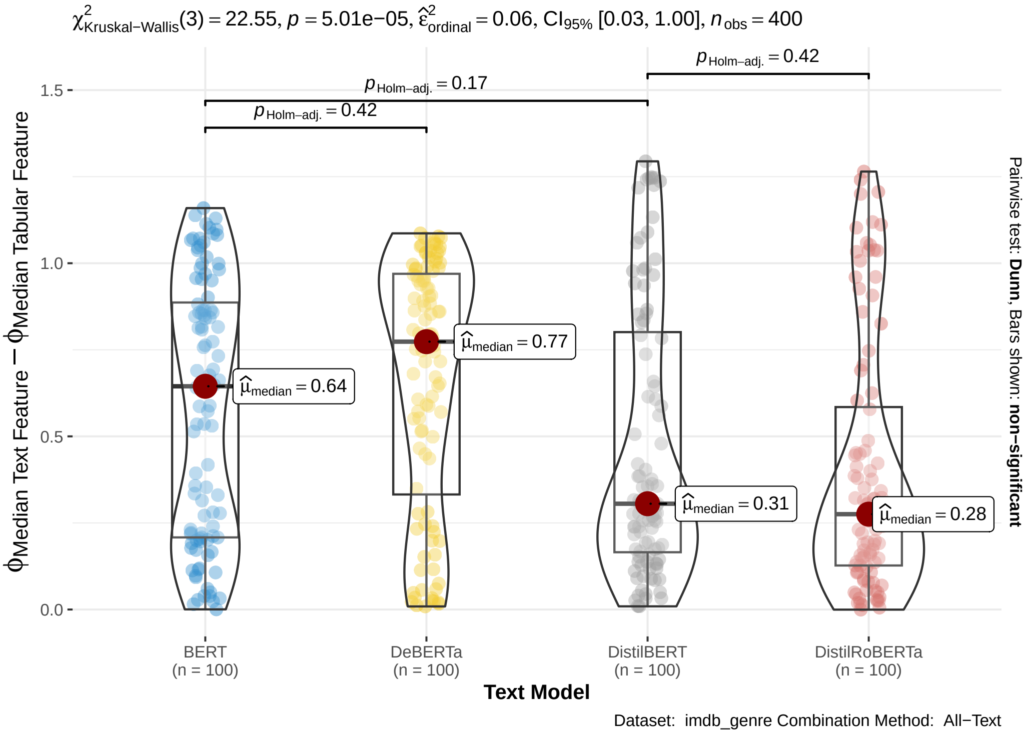


$\chi^2_{\text{Kruskal-Wallis}}(3) = 76.02$, $p = 2.19\text{e}{-16}$, $\hat{\epsilon}^2_{\text{ordinal}} = 0.19$, $\text{CI}_{95\%}$ [0.14, 1.00], $n_{\text{obs}} = 400$
$p_{\text{Holm-adj.}} = 0.10$
$p_{\text{Holm-adj.}} = 0.13$
$\phi_{\text{Median Text Feature}} - \phi_{\text{Median Tabular Feature}}$
0.2
0.1
0.0
$\hat{\mu}_{\text{median}} = 0.03$
$\hat{\mu}_{\text{median}} = 0.07$
$\hat{\mu}_{\text{median}} = 0.03$
$\hat{\mu}_{\text{median}} = 0.02$
BERT (n = 100)
DeBERTa (n = 100)
DistilBERT (n = 100)
DistilRoBERTa (n = 100)
Text Model
Pairwise test: Dunn, Bars shown: non-significant
Dataset: airbnb Combination Method: All−Text

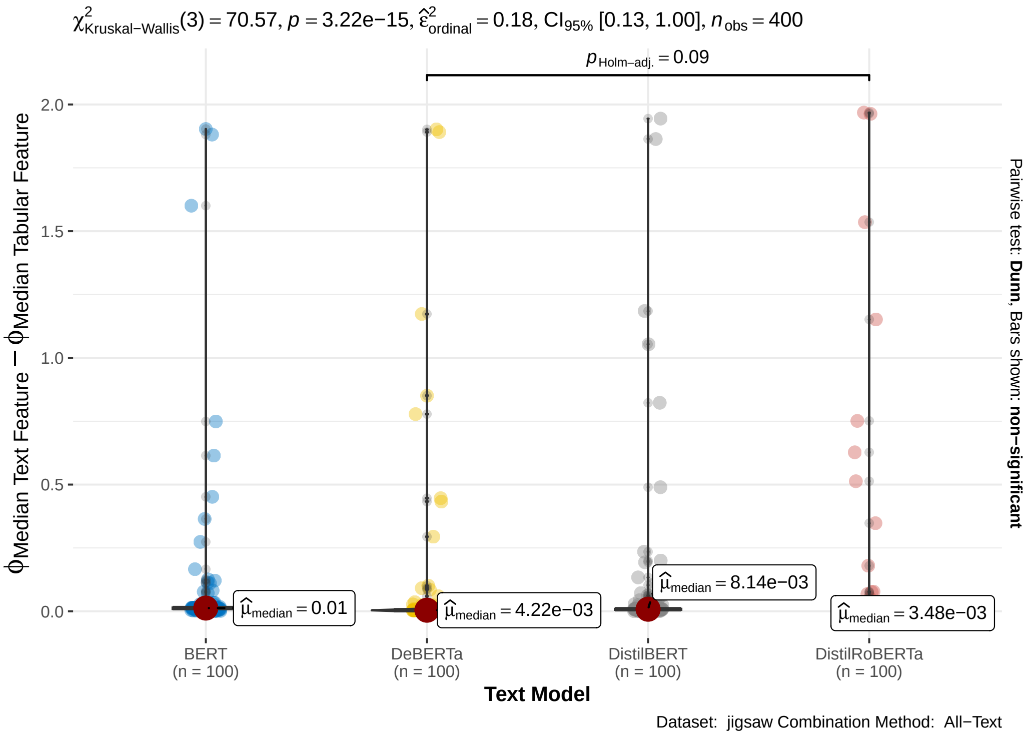
(a) Airbnb


$\chi^2_{\text{Kruskal-Wallis}}(3) = 112.32$, $p = 3.47\text{e}{-24}$, $\hat{\varepsilon}^2_{\text{ordinal}} = 0.28$, $\text{CI}_{95\%}$ [0.23, 1.00], $n_{\text{obs}} = 400$
ϕMedian Text Feature − ϕMedian Tabular Feature
1.00
0.75
0.50
0.25
0.00
−0.25
$\hat{\mu}_{\text{median}} = 0.69$
$\hat{\mu}_{\text{median}} = 0.22$
$\hat{\mu}_{\text{median}} = 0.36$
$\hat{\mu}_{\text{median}} = 0.88$
BERT (n = 100)
DeBERTa (n = 100)
DistilBERT (n = 100)
DistilRoBERTa (n = 100)
Text Model
Pairwise test: Dunn, Bars shown: non−significant
Dataset: fake Combination Method: All−Text

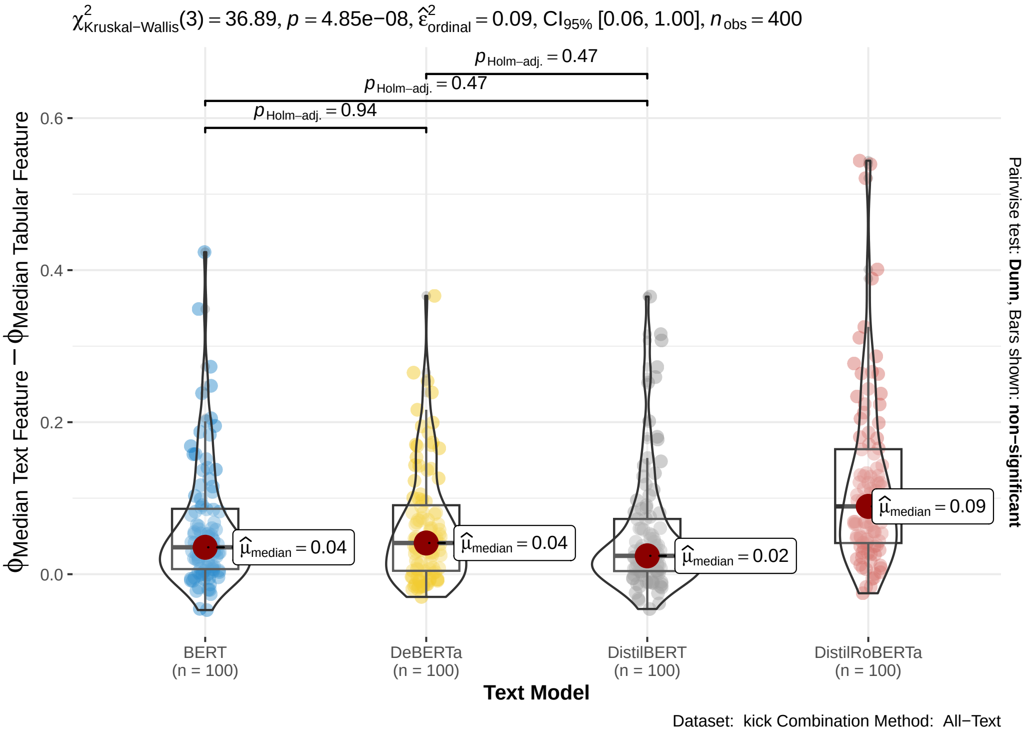
(b) Fake


$\chi^2_{\text{Kruskal-Wallis}}(3) = 22.55$, $p = 5.01\text{e}{-05}$, $\hat{\varepsilon}^2_{\text{ordinal}} = 0.06$, $\text{CI}_{95\%}$ [0.03, 1.00], $n_{\text{obs}} = 400$
$p_{\text{Holm-adj.}} = 0.42$
$p_{\text{Holm-adj.}} = 0.17$
$p_{\text{Holm-adj.}} = 0.42$
ϕMedian Text Feature − ϕMedian Tabular Feature
1.5
1.0
0.5
0.0
$\hat{\mu}_{\text{median}} = 0.64$
$\hat{\mu}_{\text{median}} = 0.77$
$\hat{\mu}_{\text{median}} = 0.31$
$\hat{\mu}_{\text{median}} = 0.28$
BERT (n = 100)
DeBERTa (n = 100)
DistilBERT (n = 100)
DistilRoBERTa (n = 100)
Text Model
Pairwise test: Dunn, Bars shown: non−significant
Dataset: imdb_genre Combination Method: All−Text

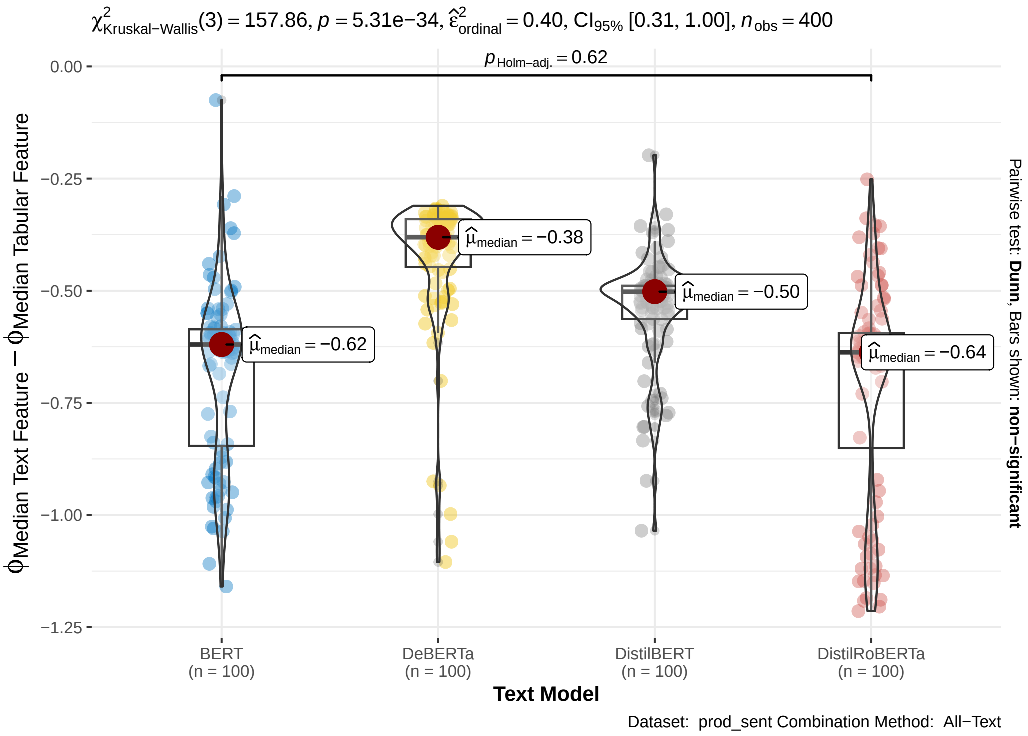
(c) Imdb



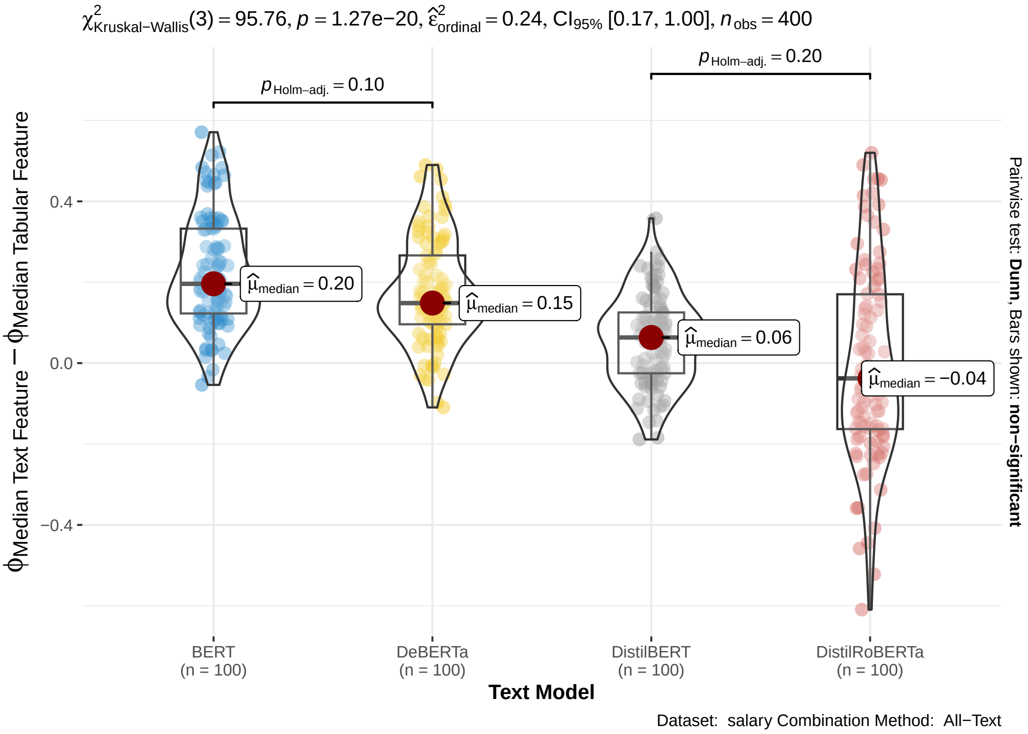
(d) Jigsaw



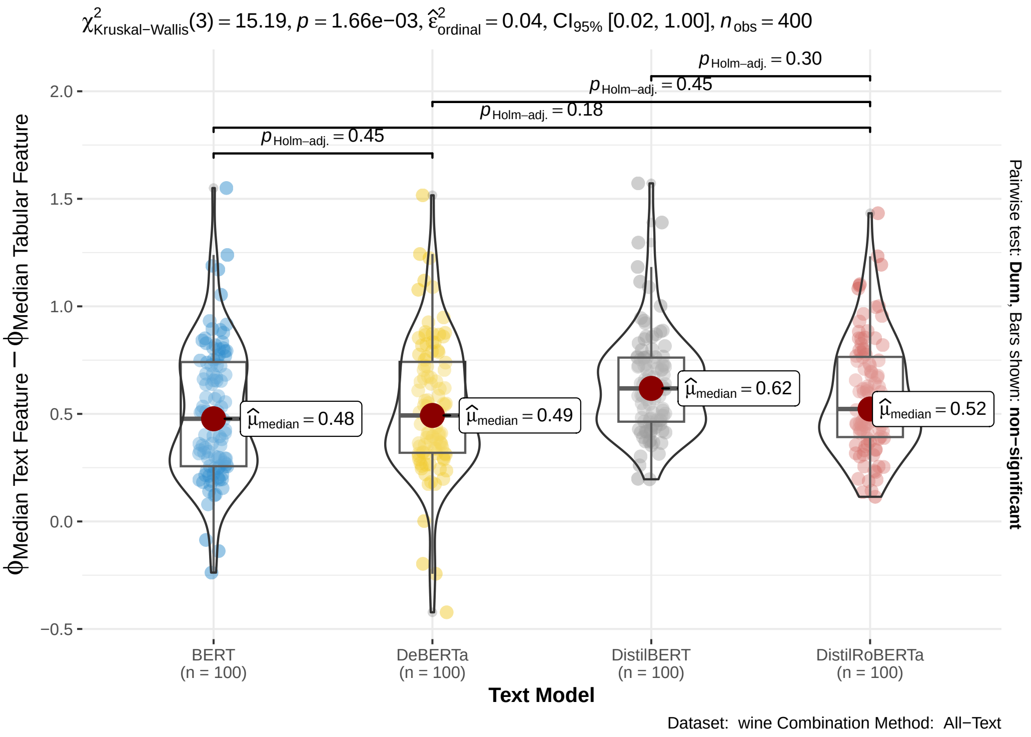
(e) Kick



(f) Prod



(g) Salary



(h) Wine

Figure 8.2: In order to compare which text model assigns more relative importance to text or tabular features, I plot $\Delta = \phi_{\text{Median Text Feature}} - \phi_{\text{Median Tabular Feature}}$on the y-axis and combination method on the x-axis for each of the nine datasets with combination method=*All-Text*. Each differently coloured plot represents the distribution of Δ's for a single text model, each with 100 coloured dots representing Δ for each of the 100 instances. For each plot, the distribution is also represented by a violin plot, which emphasises the non-normal distributions, a box plot, which indicates the spread of the data, and a labelled red dot, which indicates the median. A higher Δ indicates that a higher relative importance is assigned to text features over tabular features. The Kruskal-Wallis test statistic is significant for all nine of the datasets, meaning that the alternative hypothesis of not all medians being equal is accepted.

Bibliography

B. Alekhya and R. Sasikumar. An ensemble approach for healthcare application and diagnosis using natural language processing. *Cognitive Neurodynamics*, 16 (5):1203–1220, 2022. ISSN 18714099. doi: 10.1007/s11571-021-09758-y. URL `https://doi.org/10.1007/s11571-021-09758-y`.

J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine Bias *. In *Ethics of Data and Analytics*, pages 254–264. Auerbach Publications, 5 2022. ISBN 9781003278290. doi: 10.1201/9781003278290-37. URL `https://www.taylorfrancis.com/chapters/edit/10.1201/9781003278290-37/machine-bias-julia-angwin-jeff-larson-surya-mattu-lauren-kirchner`.

L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek. "What is Relevant in a Text Document?": An Interpretable Machine Learning Approach. *PLOS ONE*, 12(8):e0181142, 12 2016. ISSN 1932-6203. doi: 10.1371/journal.pone.0181142. URL `https://doi.org/10.1371/journal.pone.0181142https://dx.plos.org/10.1371/journal.pone.0181142http://arxiv.org/abs/1612.07843http://dx.doi.org/10.1371/journal.pone.0181142`.

A. Azri, C. Favre, N. Harbi, J. Darmont, and C. Noûs. Rumor Classification through a Multimodal Fusion Framework and Ensemble Learning. *Information Systems Frontiers*, 25(5):1795–1810, 2023. ISSN 15729419. doi: 10.1007/s10796-022-10315-z. URL `https://doi.org/10.1007/s10796-022-10315-z`.

S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PloS one*, 10(7): e0130140, 7 2015. ISSN 1932-6203. doi: 10.1371/journal.pone.0130140. URL http://www.hfsp.org/,https://dx.plos.org/10.1371/journal.pone.0130140http://www.ncbi.nlm.nih.gov/pubmed/26161953http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4498753.

S. Banerjee and A. Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

J. Bao, D. Tang, N. Duan, Z. Yan, Y. Lv, M. Zhou, and T. Zhao. Table-to-Text: Describing Table Region with Natural Language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1):5020–5027, 5 2018. ISSN 2374-3468. doi: 10.1609/aaai.v32i1.11944. URL https://ojs.aaai.org/index.php/AAAI/article/view/11944http://arxiv.org/abs/1805.11234.

A. Binder, S. Bach, G. Montavon, K.-R. Müller, and W. Samek. Layer-wise relevance propagation for deep neural network architectures. In *Information science and applications (ICISA) 2016*, pages 913–922. Springer, 2016.

F. Bodria, A. Panisson, A. Perotti, and S. Piaggesi. Explainability Methods for Natural Language Processing: Applications to Sentiment Analysis. Technical report, 2020.

T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 13-17-Augu:785–794, 2016. doi: 10.1145/2939672.2939785. URL https://github.com/dmlc/xgboost.

W. Chen, J. Chen, Y. Su, Z. Chen, and W. Y. Wang. Logical Natural Language Generation from Open-Domain Tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942, 2020a.

Z. Chen, H. Eavani, W. Chen, Y. Liu, and W. Y. Wang. Few-Shot NLG with Pre-Trained Language Model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190, 2020b.

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 1724–1734, 2014. ISBN 9781937284961. doi: 10.3115/v1/d14-1179.

K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *8th International Conference on Learning Representations, ICLR 2020*, 3 2020. URL `https://github.com/google-research/http://arxiv.org/abs/2003.10555`.

N. C. Dang, M. N. Moreno-García, and F. De la Prieta. Sentiment analysis based on deep learning: A comparative study. *Electronics (Switzerland)*, 9(3):483, 3 2020. ISSN 20799292. doi: 10.3390/electronics9030483. URL `https://www.mdpi.com/2079-9292/9/3/483/htmhttps://www.mdpi.com/2079-9292/9/3/483`.

M. Danilevsky, K. Qian, R. Aharonov, Y. Katsis, B. Kawas, and P. Sen. A Survey of the State of Explainable AI for Natural Language Processing. 10 2020. URL `https://xainlp2020.github.io/xainlp/http://arxiv.org/abs/2010.00711`.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the*

Conference, 1:4171–4186, 10 2018. URL `https://github.com/tensorflow/tensor2tensorhttp://arxiv.org/abs/1810.04805`.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT (1)*, 2019.

B. Dhingra, M. Faruqui, A. Parikh, M.-W. Chang, D. Das, and W. Cohen. Handling Divergent Reference Texts when Evaluating Table-to-Text Generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4884–4895, 2019.

W. Ding, M. Abdel-Basset, H. Hawash, and A. M. Ali. Explainability of artificial intelligence methods, applications and challenges: A comprehensive survey. *Information Sciences*, 615:238–292, 11 2022. ISSN 00200255. doi: 10.1016/j.ins.2022.10.013.

O. J. Dunn. Multiple Comparisons Among Means. *Journal of the American Statistical Association*, 56(293):52, 3 1961. ISSN 01621459. doi: 10.2307/2282330.

European Commission. Proposal for a Regulation of the European Parliament and of the Council Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legislative Acts. *Com(2021)*, 0106:1–108, 2021.

S. Farrell, C. Appleton, P.-J. M. Noble, and N. Al Moubayed. PetBERT: automated ICD-11 syndromic disease coding for outbreak detection in first opinion veterinary electronic health records. *Scientific Reports 2023 13:1*, 13(1): 1–14, 10 2023a. ISSN 2045-2322. doi: 10.1038/s41598-023-45155-7. URL `https://www.nature.com/articles/s41598-023-45155-7`.

S. Farrell, P.-J. M. Noble, and N. Al Moubayed. Natural Language Processing for Forecasting Mortality and Premature Death in Companion Animals. *Research Square*, 8 2023b. doi: 10.21203/RS.3.RS-3256060/V1.

URL `https://www.researchsquare.comhttps://www.researchsquare.com/article/rs-3256060/v1`.

F. Galton. Vox populi. *Nature*, 75(1949):450–451, 1907. ISSN 00280836. doi: 10.1038/075450a0.

E. Goldberg, N. Driedger, and R. I. Kittredge. Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53, 1994.

L. Grinsztajn, E. Oyallon, and G. Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In *Advances in Neural Information Processing Systems*, volume 35, 7 2022. ISBN 9781713871088. URL `https://arxiv.org/abs/2207.08815v1`.

K. Gu and A. Budhkar. Multimodal-Toolkit: A Package for Learning on Tabular and Text Data with Transformers. *Multimodal Artificial Intelligence, MAI Workshop 2021 - Proceedings of the 3rd Workshop*, pages 69–73, 2021. doi: 10.18653/v1/2021.maiworkshop-1.10. URL `https://git.io/J05a6`.

S. Gupta and S. K. Gupta. Abstractive summarization: An overview of the state of the art, 5 2019. ISSN 09574174.

P. He, X. Liu, J. Gao, and W. Chen. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. *ICLR 2021 - 9th International Conference on Learning Representations*, 6 2020. URL `https://arxiv.org/abs/2006.03654v6http://arxiv.org/abs/2006.03654`.

P. He, J. Gao, and W. Chen. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing. 11 2021a. URL `http://arxiv.org/abs/2111.09543`.

P. He, X. Liu, J. Gao, and W. Chen. Deberta: Decoding-Enhanced Bert With Disentangled Attention. *ICLR 2021 - 9th International Conference on Learning Representations*, 2021b. URL `https://github.com/microsoft/DeBERTa`.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. ISSN 08997667. doi: 10.1162/neco.1997.9.8.1735.

N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter. TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second. 7 2022. URL `https://github.com/automl/TabPFN.http://arxiv.org/abs/2207.01848`.

A. M. Hoyle, A. Marasović, and N. A. Smith. Promoting Graph Awareness in Linearized Graph-to-Text Generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 944–956, Online, 8 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.82. URL `https://aclanthology.org/2021.findings-acl.82`.

P. Johri, S. K. Khatri, A. T. Al-Taani, M. Sabharwal, S. Suvanov, and A. Kumar. Natural Language Processing: History, Evolution, Application, and Future Work. In *Lecture Notes in Networks and Systems*, volume 167, pages 365–375. Springer Science and Business Media Deutschland GmbH, 2021. ISBN 9789811597114. doi: 10.1007/978-981-15-9712-1{_}31. URL `https://link.springer.com/chapter/10.1007/978-981-15-9712-1_31`.

G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Y. Liu. LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 2017-Decem:3147–3155, 2017. ISSN 10495258. URL `https://github.com/Microsoft/LightGBM.`

M. G. Kendall. A New Measure of Rank Correlation. *Biometrika*, 30(1/2):81, 6 1938. ISSN 00063444. doi: 10.2307/2332226.

T. Khaleghi, A. Murat, and S. Arslanturk. A tree based approach for multi-class classification of surgical procedures using structured and unstructured data. *BMC Medical Informatics and Decision Making*, 21(1):1–12, 12 2021. ISSN 14726947. doi: 10.1186/S12911-021-01665-W/TABLES/2. URL `https:`

//link.springer.com/articles/10.1186/s12911-021-01665-whttps://link.springer.com/article/10.1186/s12911-021-01665-w.

W. H. Kruskal and W. A. Wallis. Use of Ranks in One-Criterion Variance Analysis. *Journal of the American Statistical Association*, 47(260):583–621, 12 1952. ISSN 1537274X. doi: 10.1080/01621459.1952.10483441. URL http://www.tandfonline.com/doi/abs/10.1080/01621459.1952.10483441.

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020.

C.-Y. Lin. ROUGE: A Package for Automatic Evaluation of Summaries, 2004. URL https://aclanthology.org/W04-1013.

T. Liu, K. Wang, L. Sha, B. Chang, and Z. Sui. Table-to-text generation by structure-aware seq2seq learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. 2019. URL http://arxiv.org/abs/1907.11692.

S. S. Lloyd and J. P. Rissing. Physician and Coding Errors in Patient Records. *JAMA: The Journal of the American Medical Association*, 254(10):1330–1336, 9 1985. ISSN 15383598. doi: 10.1001/JAMA.1985.03360100080018.

S. Lundberg and S.-I. Lee. A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems*, 2017-Decem:4766–4775, 5 2017a. ISSN 10495258. URL https://github.com/slundberg/shaphttp://arxiv.org/abs/1705.07874.

S. M. Lundberg and S.-I. Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017b.

Y. Lyu, P. P. Liang, Z. Deng, R. Salakhutdinov, and L. P. Morency. DIME: Fine-grained Interpretations of Multimodal Models via Disentangled Local Explanations. *AIES 2022 - Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, pages 455–467, 3 2022. doi: 10.48550/arxiv.2203.02013. URL `https://arxiv.org/abs/2203.02013v1`.

M. Mager, R. Fernandez Astudillo, T. Naseem, M. A. Sultan, Y.-S. Lee, R. Florian, and S. Roukos. {GPT}-too: A Language-Model-First Approach for {AMR}-to-Text Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852, Online, 7 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.167. URL `https://aclanthology.org/2020.acl-main.167`.

C. J. Mcdonald. The Barriers to Electronic Medical Record Systems and How to Overcome Them. *Journal of the American Medical Informatics Association*, 4 (3):213, 1997. ISSN 10675027. doi: 10.1136/JAMIA.1997.0040213.

T. Miller. Explanation in Artificial Intelligence: Insights from the Social Sciences. 8 2018. URL `http://arxiv.org/abs/1706.07269`.

S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao. Deep Learning-Based Text Classification, 4 2021. ISSN 15577341. URL `https://dl.acm.org/doi/10.1145/3439726`.

C. Molnar. Interpretable Machine Learning. A Guide for Making Black Box Models Explainable. *Book*, page 247, 2020. URL `https://christophm.github.io/interpretable-ml-book`.

A. Moryossef, Y. Goldberg, and I. Dagan. Step-by-Step: Separating Planning from Realization in Neural Data-to-Text Generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277, 2019.

R. K. Mothilal, A. Sharma, and C. Tan. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. 2020. doi: 10.1145/3351095.3372850. URL `https://doi.org/10.1145/3351095.3372850`.

C. Murawski and P. Bossaerts. How Humans Solve Complex Problems: The Case of the Knapsack Problem. *Scientific Reports 2016 6:1*, 6(1):1–10, 10 2016. ISSN 2045-2322. doi: 10.1038/srep34851. URL `https://www.nature.com/articles/srep34851`.

D. W. Opitz and J. W. Shavlik. Generating Accurate and Diverse Members of a Neural-Network Ensemble. *NIPS 1995: Proceedings of the 8th International Conference on Neural Information Processing Systems*, pages 535–541, 1995.

C. Panigutti, R. Hamon, I. Hupont, D. Fernandez Llorca, D. Fano Yela, H. Junklewitz, S. Scalzo, G. Mazzini, I. Sanchez, J. Soler Garrido, and E. Gomez. The role of explainable AI in the context of the AI Act. In *ACM International Conference Proceeding Series*, number 23, pages 1139–1150, 2023. ISBN 9781450372527. doi: 10.1145/3593013.3594069. URL `https://doi.org/10.1145/3593013.3594069`.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. ACL, 2002.

L. Parcalabescu and A. Frank. MM-SHAP: A Performance-agnostic Metric for Measuring Multimodal Contributions in Vision and Language Models &amp; Tasks. In *Proceedings of the 61st Annual Meeting of the Association for Com-*

putational Linguistics (Volume 1: Long Papers), pages 4032–4059, Stroudsburg, PA, USA, 12 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.223. URL `https://github.http://arxiv.org/abs/2212.08158https://aclanthology.org/2023.acl-long.223`.

A. Parikh, X. Wang, S. Gehrmann, M. Faruqui, B. Dhingra, D. Yang, and D. Das. ToTTo: A Controlled Table-To-Text Generation Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, 2020.

B. Peng, C. Zhu, C. Li, X. Li, J. Li, M. Zeng, and J. Gao. Few-shot Natural Language Generation for Task-Oriented Dialog. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 172–182, 2020.

R. Poyiadzi, K. Sokol, R. Santos-Rodriguez, T. De Bie, and P. Flach. FACE: Feasible and actionable counterfactual explanations. *AIES 2020 - Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350, 2 2020. doi: 10.1145/3375627.3375850. URL `https://dl.acm.org/doi/10.1145/3375627.3375850`.

R. Puduppully and M. Lapata. Data-to-text Generation with Macro Planning. *Transactions of the Association for Computational Linguistics*, 9:510–527, 2 2021. ISSN 2307387X. doi: 10.1162/TACL{_}A{_}00381/101876/DATA-TO-TEXT-GENERATION-WITH-MACRO-PLANNING. URL `https://dx.doi.org/10.1162/tacl_a_00381http://arxiv.org/abs/2102.02723`.

R. Puduppully, L. Dong, and M. Lapata. Data-to-text generation with content selection and planning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6908–6915, 2019.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, and others. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

E. Reiter. A Structured Review of the Validity of BLEU. *Computational Linguistics*, 44(3):393–401, 9 2018. ISSN 0891-2017. doi: 10.1162/COLI{_}A{_}00322. URL `https://dx.doi.org/10.1162/coli_a_00322`.

E. Reiter and R. Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997.

M. T. Ribeiro, S. Singh, and C. Guestrin. "Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 13-17-Augu, pages 1135–1144, New York, NY, USA, 8 2016a. ACM. ISBN 9781450342322. doi: 10.1145/2939672.2939778. URL `https://dl.acm.org/doi/10.1145/2939672.2939778`.

M. T. Ribeiro, S. Singh, and C. Guestrin. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016b.

M. T. Ribeiro, S. Singh, and C. Guestrin. Anchors: High-Precision Model-Agnostic Explanations. In *AAAI*, pages 1527–1535, 2018. URL `https://ieeexplore.ieee.org/document/8466590/`.

S. T. Rosenbloom, J. C. Denny, H. Xu, N. Lorenzi, W. W. Stead, and K. B. Johnson. Data from clinical notes: a perspective on the tension between structure and flexible documentation. *Journal of the American Medical Informatics Association*, page 313–317, 2017. doi: 10.1136/jamia.2010.007237. URL `https://academic.oup.com/jamia/article/18/2/181/802561`.

M. Roser, E. Ortiz-Ospina, and H. Ritchie. Life expectancy. *Our World in Data*, 2013. https://ourworldindata.org/life-expectancy.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Internal Representations by Error Propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*, page 318–362. MIT Press, 1986. ISBN 026268053X. URL `https://linkinghub.elsevier.com/retrieve/pii/B9781483214467500352`.

O. Sagi and L. Rokach. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4), 7 2018. ISSN 1942-4787. doi: 10.1002/widm.1249. URL `https://doi.org/10.1002/widm.1249https://wires.onlinelibrary.wiley.com/doi/10.1002/widm.1249`.

F. Sánchez-Vizcaíno, P. H. Jones, T. Menacere, B. Heayns, M. Wardeh, J. Newman, A. D. Radford, S. Dawson, R. Gaskell, P. J. Noble, S. Everitt, M. J. Day, and K. McConnell. Small animal disease surveillance. *Veterinary Record*, 177(23):591–594, 12 2015. ISSN 2042-7670. doi: 10.1136/VR.H6174. URL `https://onlinelibrary.wiley.com/doi/full/10.1136/vr.h6174https://onlinelibrary.wiley.com/doi/abs/10.1136/vr.h6174https://bvajournals.onlinelibrary.wiley.com/doi/10.1136/vr.h6174`.

V. Sanh, L. Debut, J. Chaumond, and T. Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.0, 10 2019. doi: https://doi.org/10.48550/arXiv.1910.01108. URL `http://arxiv.org/abs/1910.01108`.

T. Sellam, D. Das, and A. Parikh. BLEURT: Learning Robust Metrics for Text Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Stroudsburg, PA, USA, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.704. URL `http://github.com/google-research/https://www.aclweb.org/anthology/2020.acl-main.704`.

L. S. Shapley. *A Value for N-Person Games.* RAND Corporation, 4 1952. doi: 10.7249/P0295. URL `https://www.rand.org/pubs/papers/P295.html`.

X. Shi, J. Mueller, N. Erickson, M. Li, and A. J. Smola. Benchmarking Multimodal AutoML for Tabular Data with Text Fields. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 1, 11 2021. URL `https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/9bf31c7ff062936a96d3c8bd1f8f2ff3-Paper-round2.pdf`.

B. Shin, J. Hogan, A. B. Adams, R. J. Lynch, R. E. Patzer, and J. D. Choi. Multimodal ensemble approach to incorporate various types of clinical notes for predicting readmission. In *2019 IEEE EMBS International Conference on Biomedical and Health Informatics, BHI 2019 - Proceedings*, 2019. ISBN 9781728108483. doi: 10.1109/BHI.2019.8834640. URL `https://github.com/elitcloud/elit`.

R. Shwartz-Ziv and A. Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 6 2022. ISSN 15662535. doi: 10.1016/j.inffus.2021.11.011. URL `https://arxiv.org/abs/2106.03253v2`.

C. Simoiu, C. Sumanth, A. Mysore, and S. Goel. Studying the “Wisdom of Crowds” at Scale. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, 7:171–179, 2019. ISSN 2769-1330. doi: 10.1609/hcomp.v7i1.5271. URL `www.aaai.org`.

K. Simonyan, A. Vedaldi, and A. Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *2nd International Conference on Learning Representations, ICLR 2014 - Workshop Track Proceedings*, 12 2013. URL `http://code.google.com/p/cuda-convnet/http://arxiv.org/abs/1312.6034`.

W. C. Sleeman, R. Kapoor, and P. Ghosh. Multimodal Classification: Current Landscape, Taxonomy and Future Directions. *ACM Computing Surveys*, 55(7), 9 2021. ISSN 15577341. doi: 10.1145/3543848. URL `https://arxiv.org/abs/2109.09020v1`.

M. Strauss and M. Kipp. Eric: a generic rule-based framework for an affective embodied commentary agent. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, pages 97–104, 2008.

Y. Su, Z. Meng, S. Baker, and N. Collier. Few-Shot Table-to-Text Generation with Prototype Memory. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 910–917, 2021.

L. H. Suadaa, H. Kamigaito, K. Funakoshi, M. Okumura, and H. Takamura. Towards Table-to-Text Generation with Numerical Reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1451–1465, Stroudsburg, PA, USA, 2021a. Association for Computational Linguistics. ISBN 9781954085527. doi: 10.18653/v1/2021.acl-long.115. URL `https://aclanthology.org/2021.acl-long.115`.

L. H. Suadaa, H. Kamigaito, K. Funakoshi, M. Okumura, and H. Takamura. Towards table-to-text generation with numerical reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1451–1465, 2021b.

M. Sundararajan, A. Taly, and Q. Yan. Axiomatic Attribution for Deep Networks. *34th International Conference on Machine Learning, ICML 2017*, 7:5109–5118, 3 2017a. URL `http://arxiv.org/abs/1703.01365`.

M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017b.

C. van der Lee, E. Krahmer, and S. Wubben. PASS: A Dutch data-to-text system for soccer, targeted towards specific audiences. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 95–104, 2017.

J. van der Waa, E. Nieuwburg, A. Cremers, and M. Neerincx. Evaluating XAI: A comparison of rule-based and example-based explanations. *Artificial Intelligence*, 291:103404, 2021. ISSN 00043702. doi: 10.1016/j.artint.2020.103404. URL `www.elsevier.com/locate/artintCCBYlicensehttp://creativecommons.org/licenses/by/4.0/`.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need. *Advances in Neural Information Processing Systems*, 2017-Decem:5999–6009, 6 2017. ISSN 10495258. URL `http://arxiv.org/abs/1706.03762`.

G. Vilone and L. Longo. Notions of explainability and evaluation approaches for explainable artificial intelligence. *Information Fusion*, 76:89–106, 2021. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2021.05.009. URL `https://www.sciencedirect.com/science/article/pii/S1566253521001093`.

F. Wang and C. Rudin. Falling Rule Lists. *Journal of Machine Learning Research*, 38:1013–1022, 11 2014. ISSN 15337928. URL `https://arxiv.org/abs/1411.5899v3http://arxiv.org/abs/1411.5899`.

M. Watson, B. A. S. Hasan, and N. A. Moubayed. Agree to Disagree: When Deep Learning Models with Identical Architectures Produce Distinct Explanations. *Proceedings - 2022 IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022*, pages 1524–1533, 5 2022. doi: 10.1109/WACV51458.2022.00159. URL `https://arxiv.org/abs/2105.06791v2`.

T.-H. Wen, M. Gasic, N. Mrksic, P.-h. Su, D. Vandyke, and S. J. Young. Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems. In *EMNLP*, 2015.

D. H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1 1992. ISSN 08936080. doi: 10.1016/S0893-6080(05)80023-1. URL `https://linkinghub.elsevier.com/retrieve/pii/S0893608005800231`.

S. K. M. Yi, M. Steyvers, M. D. Lee, and M. J. Dry. The Wisdom of the Crowd in Combinatorial Problems. *Cognitive Science*, 36(3):452–470, 4 2012. ISSN 03640213. doi: 10.1111/J.1551-6709.2011.01223.X.

P. Yin, G. Neubig, W. T. Yih, and S. Riedel. TABERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, 2020. ISBN 9781952148255. doi: 10.18653/v1/2020.acl-main.745. URL `http://fburl.com/TaBERT`.

S. A. Zafirah, A. M. Nur, S. E. W. Puteh, and S. M. Aljunid. Potential loss of revenue due to errors in clinical coding during the implementation of the Malaysia diagnosis related group (MY-DRG®) Casemix system in a teaching hospital in Malaysia. *BMC Health Services Research*, 18(1): 1–11, 1 2018. ISSN 14726963. doi: 10.1186/S12913-018-2843-1/TABLES/6. URL `https://bmchealthservres.biomedcentral.com/articles/10.1186/s12913-018-2843-1`.

I. Zeki, Y. Hervé, J. Kan, C. Facebook, A. M. Paluri, and D. Mahajan. Billion-scale semi-supervised learning for image classification. *arXiv preprint*, 5 2019. doi: 10.48550/arXiv.1905.00546. URL `https://arxiv.org/abs/1905.00546v1`.

T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating Text Generation With Bert. In *8th International Conference on Learning Representations, ICLR 2020*. International Conference on Learning Representations, ICLR, 4 2020. URL `https://arxiv.org/abs/1904.09675v3`.

Colophon

This thesis is based on a template developed by Matthew Townson and Andrew Reeves. It was typeset with LaTeX 2_{ε}. It was created using the *memoir* package, maintained by Lars Madsen, with the *madsen* chapter style. The font used is Latin Modern, derived from fonts designed by Donald E. Kunith.