Impacts of Model Choice in XAI

by

Graham Alexander

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Explainable artificial intelligence models are becoming increasingly important as restrictions grow for corporate use of blackbox models whose predictions affect people's lives and yet cannot be interpreted. Black boxes do not convey trust to end-users and are difficult to train and debug for developers.

Model agnostic explanation methods, like SHAP [23], can be used post hoc to shed light on these blackbox predictions. With access to a model's predictions, SHAP can generate scores for relative feature importance. This work focuses on explanations generated for Natural Language Processing (NLP) where the features that SHAP uses are words.

There are currently no generally accepted methods to generate explanations in NLP. However, SHAP can calculate importance scores for each word where the most important words can be taken as the explanation. SHAP should be structure-agnostic, meaning it should not be influenced by the number or types of layers in the model, it should only be influenced by the quality of the predition. Otherwise, SHAP predictions cannot be fairly compared across models because SHAP may be biased towards certain structures.

Importance scores from SHAP are converted to a mask to either include or ignore each word of the input, providing the generated explanation. The Eraser [10] dataset provides human annotated explanations for NLP tasks that can be used as a gold standard by comparing them to the explanations generated by SHAP. An F1 score can then be used as a notion of the quality of the explanation by comparing the generated explanation to the human annotated

explanation.

This work investigates whether the quality of explanations generated by SHAP is structure agnostic. Using a dataset with ground truth explanations in a sentiment analysis task, we compare the SHAP output across different types of models. Our main finding is that CNN models using intrinsic explanation underperformed CNN models without intrinsic explanation, while having nearly identical accuracy. These findings demonstrate that the underlying model can impact SHAP's performance and may favour certain structures of models.

I asked myself the other day: is it the pants that don't fit me anymore? or is it me who doesn't fit into my pants now? it's a very important question because the answers determine the course of action: go to a tailor or go to a gym.

– Aksakal on stack overflow

Acknowledgements

I have been helped by so many people along the way, and this is a list that continues to grow the longer I think about it. For giving me the freedom to explore ideas and the patience to let me see them through, I'd like to thank my supervisor Dr. Denilson Barbosa. While my topic ended up being tangential to areas he had worked in before, I was always encouraged to develop these ideas. I'd also like to thank my brother, Brendan, who manages to make statistics seem engaging. I talked through many statistical hypotheticals with him over weekend BBQs. My parents have been strong proponents of education throughout my life, and I'm lucky that it finally managed to click for me so that I could return to finish my studies. This return to school was made much more fulfilling by professors like Dr. Nicholas Boers who showed such a love for the subject and expected more of his students.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

Machine learning (ML) models often differ from statistical models in that the former focus on prediction, while the latter focus on inference [16]. When focusing on prediction, different models are fit to maximize some sort of measure of accuracy, without needing to understand why a prediction was made. While for inference, different models are fit to the data with the intent on learning about the phenomena that created the data.

Many projects have focused on prediction, which has led to progressively larger ML models. From BERT's 110 million [9] to NVIDIA MegatronLM's 8.3 billion [34] to OpenAIs GPT-3's 175 billion [5] parameter network, these highly accurate models have far too many parameters for a human to interpret. When a model's process for calculating the prediction is not understood, it is considered a blackbox.

A number of different approaches have been designed to address this issue of explainability, as explainability is considered a positive feature of a model [12]. A correct answer may seem confusing without the supporting evidence, and an incorrect answer may be easier to spot if it is paired with the faulty explanation. By providing explanations, whether the provided prediction is correct or not, the model can be more readily trusted.

Intuitively, an explanation should reflect the reasoning that led to the answer [17]. The "answer" in the case of an ML model is a prediction based on some input data. The explanation is supporting evidence for why a prediction

was made. Given this picture (input), it is not a picture of a cat (prediction); the object had feathers (explanation). This work considers two approaches used to address the issue of black box models: intrinsic explanation and post hoc explanation.

Intrinsic explanations are generated by any framework where models are designed to be explainable by having some part of the output structured to be the explanation of the model. In the case of linear regression, the regression formula is considered to be the explanation. While in ML, some layer of the model needs to be trained to give the explanation. Lei *et al.* [22] designed a two-model architecture, such that the first model generates the explanation that the second model uses for prediction. We use this framework to generate the intrinsic explanations involved in the experiments in our work.

Post hoc explanations are generated by any framework that can be applied to a model. SHAP (SHapley Additive exPlanations) [23] is considered to be a model agnostic form of post hoc explanation in that it can be used as a wrapper around any previously trained model. By model agnostic, we mean that the underlying components or processes of the model should not affect the explanation; only the accuracy of the prediction need be considered.

These approaches are not mutually exclusive. A model that has been trained with intrinsic explanation could still be used in a post hoc explanation framework. By setting up an experiment where the intrinsic explanation can be removed, we can determine whether these two approaches are complementary or antagonistic.

SHAP does not have access to any of the model's intermediary weights or parameters. The explanation that SHAP produces is based solely on the relation between the input and prediction of the model. This is, to a large extent, why SHAP can be model agnostic. By gradually altering the inputs to the model and observing the changes in prediction score, SHAP can determine which features of the input were most relevant to the prediction. Features that contributed most to the prediction can be thought of as the explanation. Feature relevance, represented by a SHAP value, is then a product of the model, and any differences in these values can only be attributed to properties

of the model.

We look at training two different structures of models and comparing the resulting explanations that SHAP produces. If SHAP is model agnostic, then it should only be the predictions that affect the quality of explanations. Any differences between the quality of explanations should be random noise and not biased towards a particular underlying structure. If there are systematic differences between these structures that cannot be explained by prediction scores, then SHAP must be affected by model structure.

1.2 Motivation

1.2.1 Explanations

Simply ignoring the drawbacks of purely predictive models is no longer an option. There are cases of ML models with exagerated training scores in medical imaging that were actually just using remnants of marker left by the medical professionals that annotated the data [42]. Other cases show insignificant pertubations in the input leading to wildly inaccurate predictions [37].

We can negate errors and identify biases in models by exposing how the model arrived at its decision. The General Data Protection Regulation has started to legislate the need for transparent or explainable models in businesses [11].

There are different interpretations of explainable artificial intelligence (XAI) terms, but in general the more questions you could ask of your model and have answered, the more understandable the model. Typical questions include:

- Is a particular feature redundant?
- What if a particular feature had been a different value?
- Which feature contributed most to this decision?

1.2.2 Natural Language Processing

Natural language processing (NLP) is a domain that has an intricately dependent and vast feature space that is human language. Other domains with tabular and numerical data benefit from having the input space represented in a way that is often objective and easier to interpret. The numerical nature allows for straightforward comparisons of data points, while the tabular nature often means that the number of features will not dwarf a reasonable cognitive effort. In human language, there are subjectivity issues. Consider a sentiment analysis task [35] where we need to judge the magnitude of emotion. Now imagine having to determine which of the following statements carries the more negative sentiment: "I hate ice cream" or "Ice cream is the worst"?

To represent language we can use some one-hot encoding or bag-of-words model. One-hot encodings are vectors with as many elements as you are trying to encode. In NLP, each element of the vector corresponds to a unique word of the corpus. To encode a particular word, every element of the vector is set to zero except for the element that corresponds to your word, which is set to one. This leads to an incredibly large and sparse matrix that is no more than a lookup table. Simple methods, like one-hot encoding, are at their limits in many tasks [26]. If you choose instead to convert words or sentences to learned embeddings, like embeddings from BERT[9] or USE[6] , then you add a level of complexity to your input space which is difficult for a human to understand.

For the experiments in this work, we define an explanation in NLP in an extractive sense. An extractive explanation is some subset of the input words, that generate a near identical prediction as compared to the prediction from the entire input set. In the top sentence of fig. 1.1, the word "good" (and "terrible" in the bottom sentence) is responsible for a large fraction of the model's deviation from the base value. We can consider "good" and "terrible" as explanations for the two phrases.

Determining the number of words to include in the explanation will depend upon the sentence given. A short sentence may only have a word or two that were relevant to the prediction, while a longer passage may require far more.

Instead of choosing a fixed number, we use a threshold that is relative to the prediction. This threshold governs what percentage of the prediction the words must account for to be considered. Based on a chosen threshold, these SHAP values can then be evaluated as a string of binary variables determining whether we keep or discard the word associated with that binary variable. This string of binary variables, or mask, creates an extractive explanation from the input text.

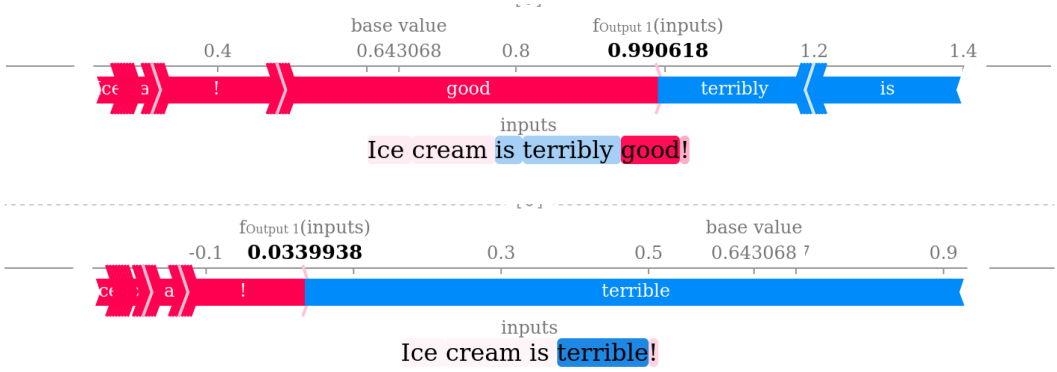

Figure 1.1: Illustration of SHAP's highlighting based on given sentence. Red highlighting contributes to a positive predition, blue highlighting a negative one.

1.2.3 Model Alteration

Model alterations are used to generate pairs of similar models that differ only by this alteration.

In linear regression with regularization, normalization of input can positively impact the results. Regularization can help with issues of multicollinearity and feature importance. Both of these issues are a concern for a sentiment analysis task [20].

Model alterations are used in our experiment to create the basic architecture, to create two different structures of models and their resulting SHAP values compared by calculating an F1 score with a set of human annotated explanations.

This thesis explores the following question:

Does the underlying structure of a model impact the resulting explanations from SHAP?

We explore this question using a sentiment analysis task in NLP. Sentiment analysis is the approach to determine the valence for a given language input. Provided some text, for example, a movie review, decide whether there is a negative (0) or positive (1) valence associated with it. Several different neural networks are used, and their decisions are fed into SHAP. SHAP then outputs the relative importance of each word towards the model's prediction.

The belief that SHAP is model agnostic prompts this investigation. SHAP can produce feature importance for any model, without needing to know anything about the inner layers, leading to this categorization of being model agnostic. However, if knowing some aspect of the inner layers, while keeping prediction accuracy consistent, can affect SHAP's output then SHAP is perhaps not as model agnostic as once thought.

Chapter 2

Background Material

In this chapter underlying concepts of this thesis that pertain to model explanations are discussed. To begin, definitions of some XAI terms are expressed. With these definitions, embeddings can be related to explanation. Next, a summarization of the types of models that are used. Different models have different ways of processing the input and may behave differently with the explanation methods. Finally, an overview of the two methods of explanation explored in this work is given.

2.1 Common Terms

Many terms in XAI are not yet concretely defined. Some terms like interpretable and explainable are used interchangeably in some papers, while differentiated in others [3]. Some terms, like interpretability and explainability, are not absolutes and are better described with relations. Regression is often a baseline to compare to, because it can often be used to generate a solution to machine learning questions.

2.1.1 Interpretability

Interpretability defines a scale of how easily understood a model is. How easily can one follow the path from the inputs to the outputs; and for layers or components themselves, how readily can we digest the information using a reasonable cognitive effort. Linear regression models are often an example given for an interpretable model because each step of the calculation has a

purpose that can be readily understood.

Many deep neural networks are seen at the other side of the interpretability spectrum, where calculations are too numerous and logic not understood. Model complexity is often considered equivalent to a lack of interpretability [27].

2.1.2 Explainability

Like interpretability, explainability also speaks to how well understood a statistical or ML model is. Explainability differs in that it is not a matter of being able to follow the calculations of the model, but understanding how or why the model arrived at a certain answer.

An explanation should highlight the relevant details for how a decision was made. Linear regression models are also explainable because they give a formula for how to predict a new datapoint. We can also ask questions, such as: "what if feature 1 had been a different value?", by simply modifying the input variable to the regression equation.

2.1.3 Explanation Scope

Local explainability refers to how the prediction of a particular instance was done. Techniques like SHAP offer local explainability, as Shapley values give feature importance for a specific datapoint that do not necessarily apply to the dataset at large.

Global explainability refers to how a model's predictions are made throughout the dataset. Regression offers a global explanation, because a common formula used to calculate all datapoints is returned.

2.1.4 Transparency

The term blackbox refers to a model that is on one extreme of the interpretability spectrum, where all internal weights, parameters, and calculations are kept hidden; only the inputs and outputs are known. The model may give fantastic results, and may even offer an explanation, however there is no expectation to

interpret the steps of the model with reasonable cognitive effort. SHAP treats all models as blackboxes, because it only accesses the input and output values.

In contrast to a blackbox, a model could be deemed a whitebox or glassbox, meaning that its internal weights and/or processes are exposed. This can allow explanation methods to be designed for such a model, taking advantage of internal parameters like attention weights. For example, Sundararajan *et al.* [38] use calls to the gradient operator to calculate what they dub "integrated gradients".

2.2 Embeddings

Embeddings are multidimensional numerical representations of objects. In NLP, these objects are word related; in this work they are numerical representations of words. Embeddings are meant to cluster semanticly similar objects even if they are syntactically distant. However the vectors are generated, they generally result in embeddings which make little or no sense to a human. Although embeddings lack interpretability, they still seem necessary for AI since they are so prominent in state-of-the-art research[5], [9], [15], [43]. They can be learned through a number of approaches like word2vec (W2V) [26] or Universal Sentence Encoder (USE) [6]. Whatever the method of construction, the process of embedding changes the objects of variable length to vectors of fixed length – a critical step to make the natural language an acceptable input for ML.

Most embeddings do not constrain the dimensions to line up with certain attributes – like a happiness dimension or a part of speech dimension. There has been work in this direction, which could help alleviate interpretability issues [28]. The only real human interpretability that embeddings give us is their relative values in terms of a similarity metric like cosine similarity.

2.3 Models

Post hoc and intrinsic explanations are tested on a variety of models as an exploration of the space. These models include convolutional neural networks

(CNN) [21], long short term memory networks (LSTM) [14], and transformer networks [40]. These models process their input quite differently and their explanations may reflect these differences.

2.3.1 CNN

CNN process data by sliding a window, called a kernel, over the input and performing a convolution operation. Convolution, in this scenario, is the element-wise multiplication and then sum of the kernel with the input. The kernel operates like an n-gram model, processing the sentence n tokens at a time, illustrated in fig. 2.1. A CNN can have many kernels of the same and differing sizes, all of which can create higher-order features for subsequent layers of the network.

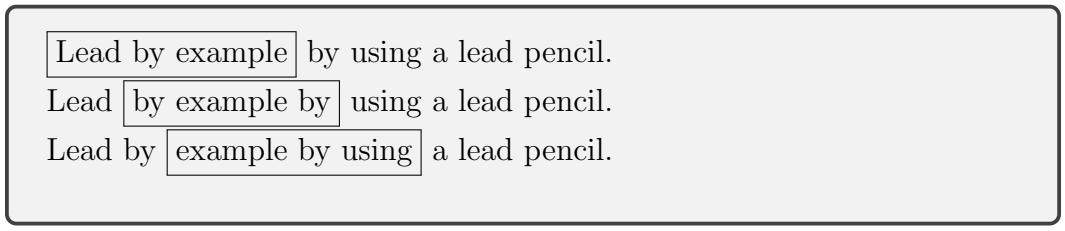

Figure 2.1: Illustration of sliding window of convolution for CNN. The 1-dimensional kernel is of size 3. The first 3 trigrams of the sentence are shown.

The different convolution operations are independent and can be performed in parallel, making training fast. CNNs possess properties that are ideal for NLP, such as local invariance and compositionality [39].

2.3.2 RNN/LSTM

Recurrent neural networks (RNN) force the input to be processed one token at a time, similar to a human's sequential reading, treating the input as a time series [39]. The sequential nature allows the RNN to bring context information calculated from previous tokens of the sentence. The context, or state, is calculated for each element of the input by combining the previous state and the current token's embedding. This state can act as a memory, allowing connections to be made between words both near and far. A layer is needed

for each token of the input, however we often think of an RNN as a single layer. This distinction is illustrated in fig. 2.2. Although the RNN seems to run similarly to a human's interpretation of a sentence, the serial nature of its processing makes it much slower than its competitors.

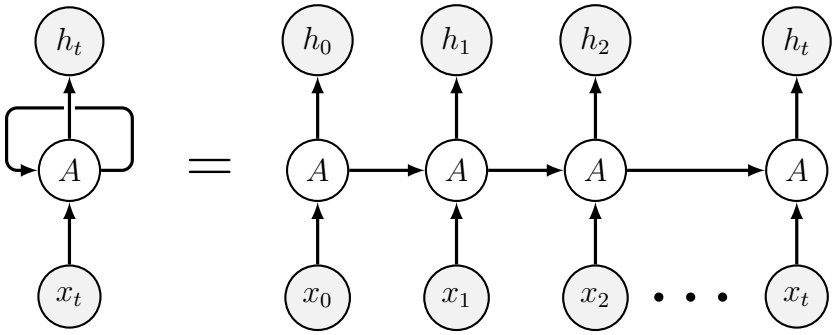

Figure 2.2: On the left: a compact illustration of the RNN. On the right: an unrolled illustration of the RNN where each layer is shown separately.

As the forward pass of an RNN processes input one token at a time, so must the backpropagation. The longer the input, the longer the chain for backpropagation in the model. When many small or large derivatives are multiplied together the backpropagation gradient can grow or shrink to an unusable term. This problem of vanishing or exploding gradients is partially addressed with long short term memory networks (LSTM). Instead of trying to represent all context information in the state, the information is regulated by a number of gates that allow the LSTM cell to add or remove information from the state. These gates allow the network to keep associations between distant points of the input and drop unimportant information without suffering from vanishing or exploding gradients.

2.3.3 Transformer

Transformers are networks built around self-attention [40]. Self-attention is the mechanism that a transformer uses to incorporate the context of the input. Three large matrices (query, key, and value) are maintained that allow the comparison of all elements of the input to be done in parallel. Each element of the input has its own query, key, and value vectors. The relation between

each element with every other element of the input is done through the dot product of the current element's query vector with each other key vector. This process allows each element to incorporate the effect of other elements through a weighted sum of their value vectors.

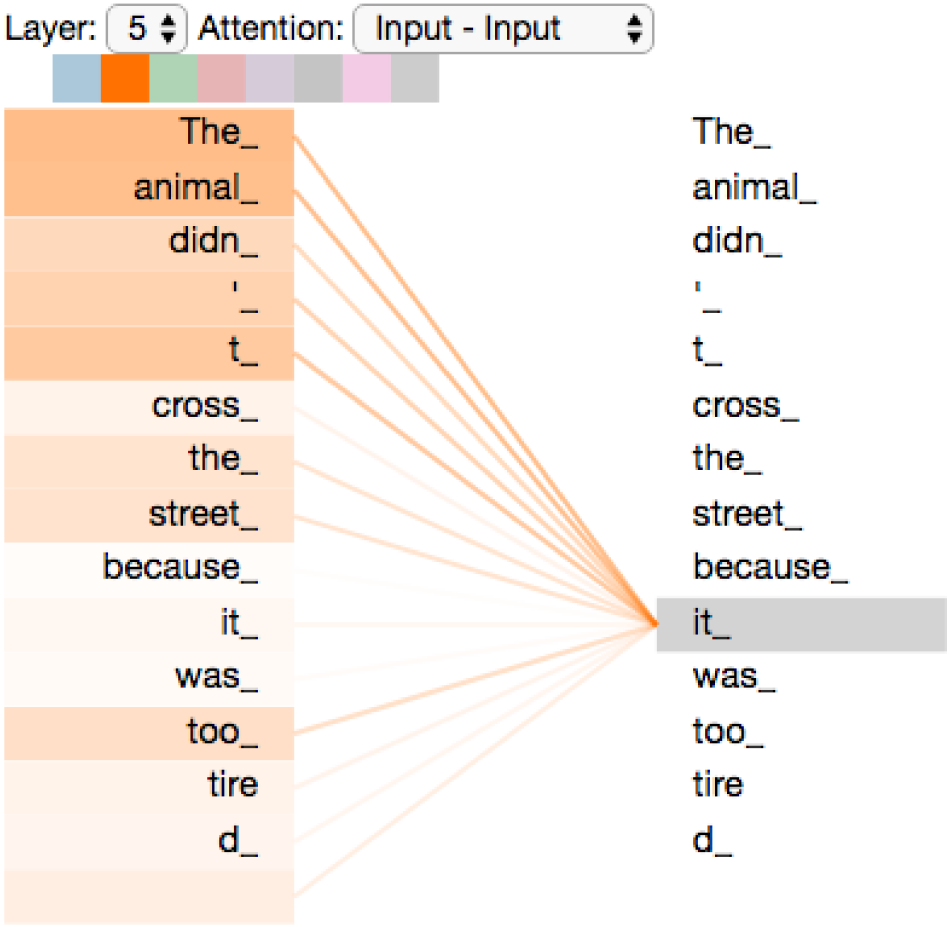

Figure 2.3: Magnitude of attention weights, shown by intensity of the tan coloured highlighting, from a transformer model[1]. "The animal" which is what "it" refers to, is part of what the attention mechanism was focusing on. Alammar, J (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar .github.io/illustrated-transformer/ and shown under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

2.4 Explanation

2.4.1 Post Hoc Explanation

Post hoc explanations can be obtained from fully trained networks, even if they were not designed with explainability in mind. Many post hoc explanation methods are model agnostic [25].

2.4.2 SHAP

SHAP (SHapley Additive exPlanations) is a method that leverages the idea of Shapley values, from coalitional game theory[33], to compute feature contribution[24]. Shapley values are the marginal contributions of each input element across the power set of possible coalitions. That is to say, for every set of input features that do not include a certain feature, what is the contribution made by adding this feature. In each set of input features, the contribution may be different, so the average contribution is taken across all sets of features.

This attribution method satisfies the properties of efficiency, symmetry, and dummy [33].

- Efficiency: The sum of all Shapley values is equal to the prediction made by the model with all features.
- Symmetry: Two features that affect the model equally in all coalitions will have identical Shapley values.
- Dummy: A feature that never affects the model will receive a Shapley value of 0.

In NLP, however, performing the calculation of marginal contribution across all coalitions of a natural language input that could easily contain hundreds of words is not feasible. Instead, SHAP approximates Shapley values by using a randomly generated subset of the possible coalitions. The original model is used to make the predictions for these new inputs, handling feature absence by sampling another random instance. Integrating over this marginal distribution gives us the Shapley approximations.

In NLP, SHAP doesn't need to sample from other instances when a feature is left out, because unlike tabular data, natural language does not have a set number of columns to fill. Instead, a particular feature that is left out is masked in such a way that the model will ignore it. Since the calculation of Shapley values can be computationally taxing, SHAP also makes use of a related concept called Owen values. Owen values are a specific scenario of the game theory situation where specific coalitions of features are not possible. In fig. 2.4, all possible combinations of the three features f_1,f_2,f_3 are listed, however, it could happen that the presence of f_3 is only possible when f_2 is already there – Owen values allow for this adaptation.

In fig. 2.5 and fig. 2.6, SHAP's initial partition graph is shown. This interactivity graph is used to calculate Owen values, which are used in the place of Shapley values.

The general formula for Owen values is:

$$\phi_i(v, B) = \sum_{R \subseteq M \setminus B_k} \sum_{T \subseteq B_k \setminus i} \frac{1}{m b_k} \frac{1}{\binom{m-1}{r}} \frac{1}{\binom{b_k-1}{t}} [v(Q \cup T \cup i) - v(Q \cup T)]$$

Where M is the set of all possible coalitions, B_k is the coalition containing i, and $Q = \cup_{r \in R} B_k$.

The calculation for the short sentence "Ice cream is great" is given in table 2.1, where the sentence had the partition scheme {{0,1},{1,2}}.

Q	T	Union(Q,T,i)	difference	weight
		ice	0.19	.25
	cream	ice cream	0.28	.25
is great		ice is great	0.00	.25
is great	cream	ice cream is great	0.00	.25

Table 2.1: Owen values calculated for the sentence "Ice cream is great"

2.4.3 Model Intrinsic explanation

In contrast to model agnostic frameworks, if the explanation is built into the model itself, it is a model instrinsic explanation. Since this type of explanation

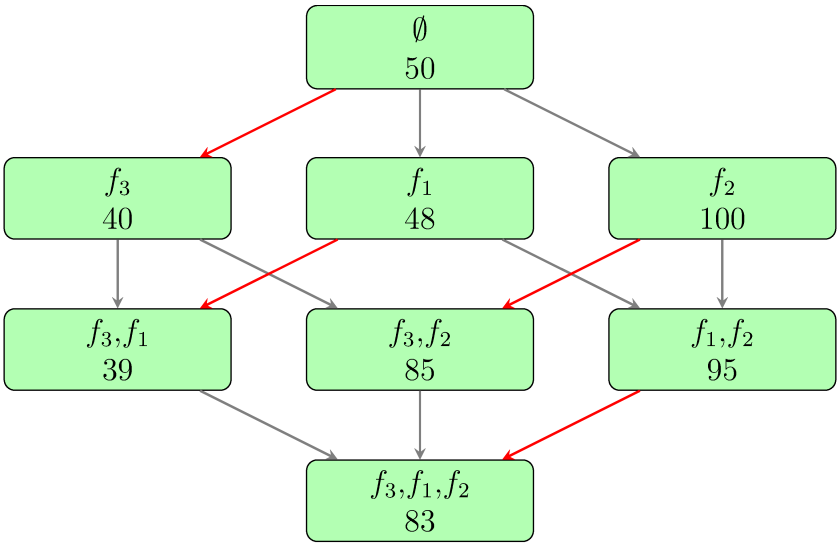

Figure 2.4: Powerset of featureset. Each node represents a unique model trained on the contained features. The integer represents the value of some sample evaluated on that particular model. The edges show the marginal contribution of a particular feature. All marginal contribution of the f_3 feature are highlighted in red.

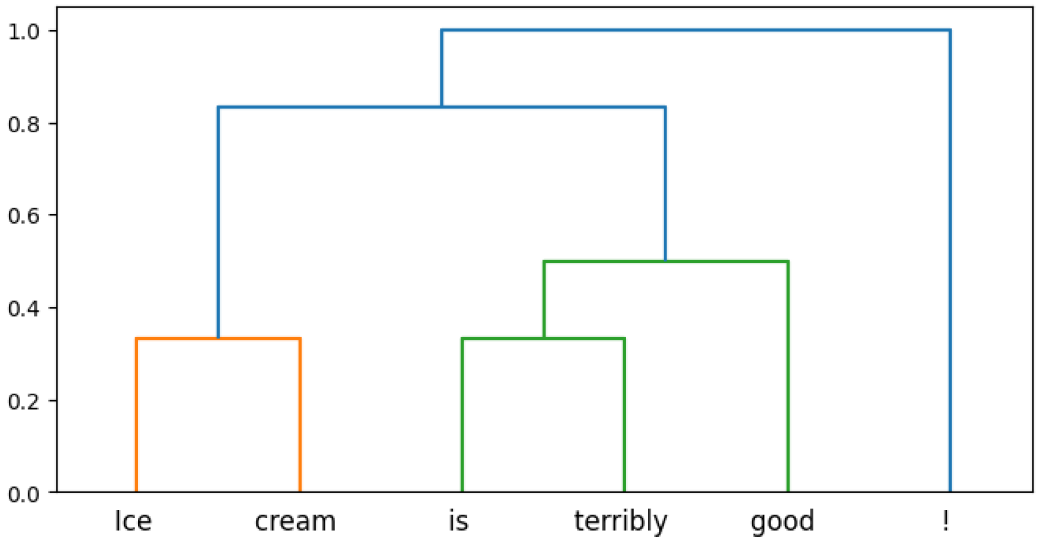

Figure 2.5: Partitioning done by SHAP for the sentence: "Ice cream is terribly good !"

is an output of the model during training, labels for the explanation could be used. In the case of Lei *et al.* [22] , no labels were used. Instead, the explanation was an extraction from the input, regulated by its length and continuity. The basic model structure is illustrated in fig. 2.7. This model was

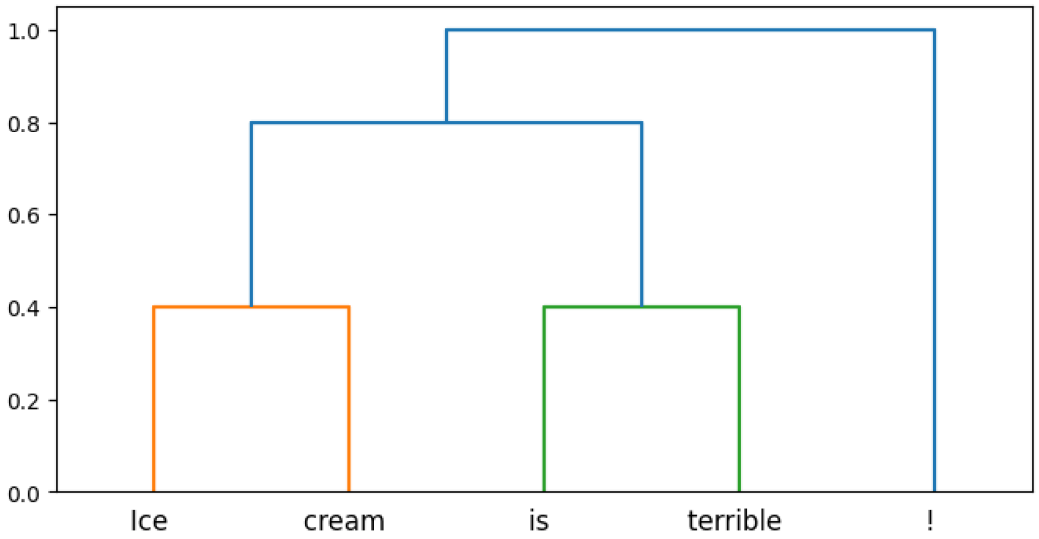

Figure 2.6: Partioning done by SHAP for the sentence: "Ice cream is terrible !"]

sentence	prediction result	shap values
[Ice, cream, is, terribly, good, !]	.99	[.05, .05, -.24, -.24, .67, .05]
[Ice, cream, is, terrible, !]	.03	[.06, .07, .06, -.87, .06]

Table 2.2: SHAP values for each word sum to the prediction result when added to the base score, in this case .64

designed for the NLP task of sentiment analysis. It obtains its explanation by breaking up the model into a generator and encoder.

The loss function for the network is as follows:

$$\mathcal{L}(z, x, y) = E(enc(z, x), y)$$

In words: the input x and generator output z is given to the encoder module (enc). A binary cross entropy E is then calculated between the encoder's output and the label.

While the loss function for the generator is:

$$\Omega(z) = \lambda_1 ||z|| + \lambda_2 \sum_t |z_t - z_{t-1}|$$

In words: the first term evaluates the generator's output z on the number of tokens included, while the second term evaluates for contiguity by checking how often the t^{th} term of z differs from the $t + 1^{th}$.

The generator's task creates a mask by choosing to keep (1) or exclude (0) each word in the input, while the encoder predicts the sentiment of this masked input. The generator is governed by a loss function that grades the mask it produces in terms of length and continuity. The length term ($\lambda_1||z||$) is the sum of the original mask, giving a total of the included elements. The continuity term ($\lambda_2 \sum_t |z_t - z_{t-1}|$) is then calculated by adding the absolute value of the difference between each time step, giving a measurement of how often the mask oscillates. The goal is to minimize these two terms, which will ideally give short, contiguous phrases.

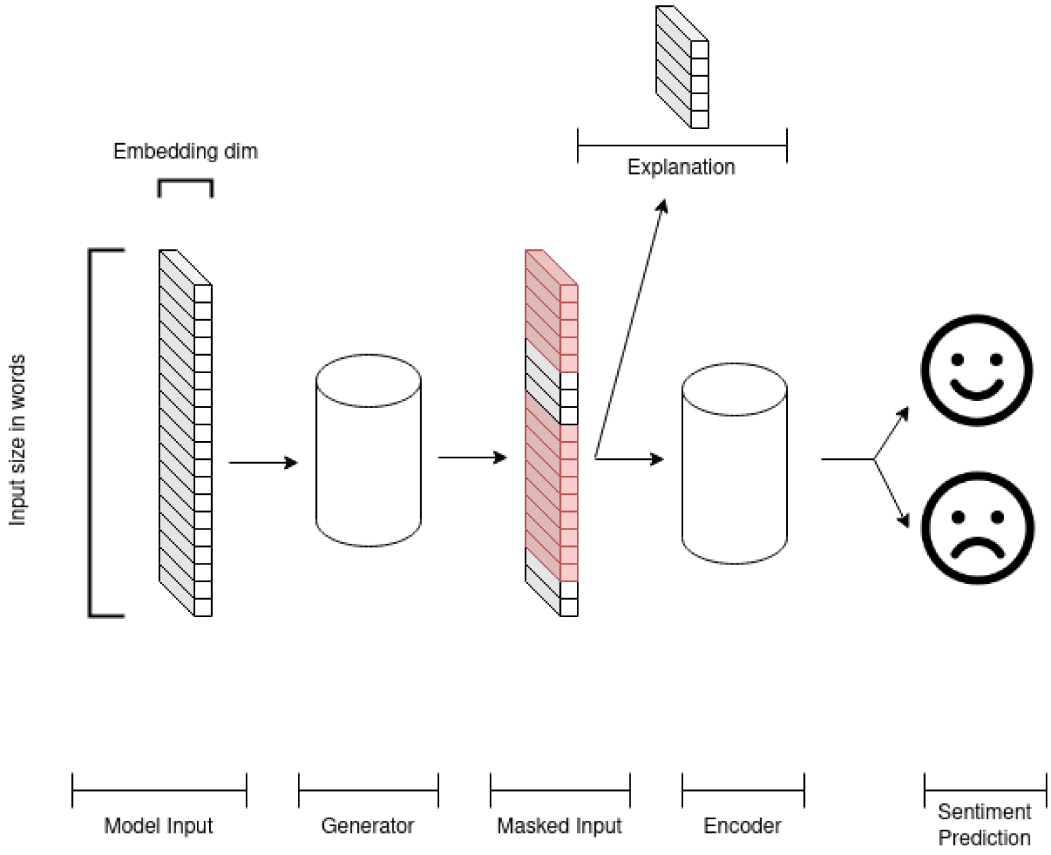

Figure 2.7: Intrinsic explanation model setup. The red shading indicates words that have been masked to zeros and will be ignored by the encoder layer.

Chapter 3

Related work

In this chapter, two alternatives to SHAP are introduced. Next, a discussion on whether using a model's latent layers constitutes a reasonable explanation is given. Finally, a description of a method that extends SHAP values into the causal domain concludes the chapter.

3.1 Alternatives to SHAP

In this work, SHAP was used because it is a very popular explanation framework. However there were other candidates. This work only requires that the explanations are computed post-hoc. Two other popular post-hoc explanation frameworks exist, namely, Local Interpretable Model-agnostic Explanations (LIME) [32] and Anchors [31]. Both of these methods, like SHAP, work by perturbing input data and observing how those changes affect the model's predictions.

LIME generates an explanation by building a simple surrogate model that is locally faithful. The simple surrogate model is often linear, categorizing all data as either one of two categories. While Ribeiro *et al.* [32] focus on linear models, their approach could be used with other classes of models deemed interpretable, like decision trees. LIME was shown to be a member of the additive feature attribution methods [23], meaning its explanation model was a linear function of binary variables. In NLP, LIME generates the same sort of extractive explanations that SHAP does by using this binary variable for each input token. However, unless the decision boundary between the two classes

is truly linear, LIME can only achieve local faithfulness.

Instead of creating a linear model, the Anchors framework finds particular "rules" that when held, give some predefined accuracy measured across the subset of data where this rule applies. A rule is comprised of some group of features being certain values or in certain ranges. For example, when feature 1 is greater than 5, the output is False with 98% accuracy. More succinctly, an anchor is an IF-THEN rule holding your predefined accuracy. Ribeiro *et al.* [31] also define the notion of coverage as the probability that this anchor applies to possibly unseen instances of a dataset. In NLP, Anchors' rules may return n-gram like extractive explanations, such as IF:"not bad" - THEN:predict positive.

3.2 Using model internals as explanation

Model internals, like attention weights, have garnered interest to be used as explanations for a model's predictions [2]. This interest has sparked a debate for whether the use of attention weights as an explanation is valid [18], [41]. Jain and Wallace [18] found that alternative attention weights could be used in the model that did not change the model's prediction but that did change the explanation. In addition, attention weights do not always align with gradient-based measures. Wiegreffe and Pinter [41] disputes some of these claims, arguing that the alteration of attention weights goes against what the model has learned throughout training.

While it is still unclear to what degree attention weights represent a valid explanation, this work attempts to avoid the debate by using the structure of the model as the justification for the explanation [4]. Instead of looking at unconstrained attention weights, which are a weighted contribution from many sources, the two part model used in this work first selects a subset of the input tokens and then makes a prediction based solely off this subset. This ensures that only the elements from the explanation contribute to the prediction.

3.3 Using Do-calculus in conjunction with Shapley Values

Shapley values are key in many model-agnostic explanations, however, Shapley values can lead to counterintuitive explanations when features vi¡olate the independency assumption of its features [13]. Natural language features are words and to operate as if the occurence of each word is completely independent of the rest of the sentence is incorrect. Heskes *et al.* [13] proposes using Do-calculus [29] and a causal graph to eliminate the independency assumptions that can otherwise lead to counterintuitive explanations. "Marginal Shapley Values" are produced from this independency assumption and are a method for reducing computational complexity of computing "Conditional Shapley Values". Heskes *et al.* offer a method of computing Shapley values that takes into account causal relationships in the features; they dub them "Causal Shapley Values".

To further specify these Causal Shapley Values, Heskes *et al.* decompose this total effect into a direct and an indirect effect. The direct effect is how some cause Y is affected solely by random variable $X2$, possible causal models illustrated in fig. 3.1. The indirect effect is how some other random variable, $X1$, contributes to Y. The distinction between which variable is called direct or indirect is a matter of focus in the experiment. In an experiment where you are testing the efficay of a new drug, while measuring its interaction with some lifestyle choice (e.g. amount of water drunk), one would name the drug effect as direct and the water effect as indirect.

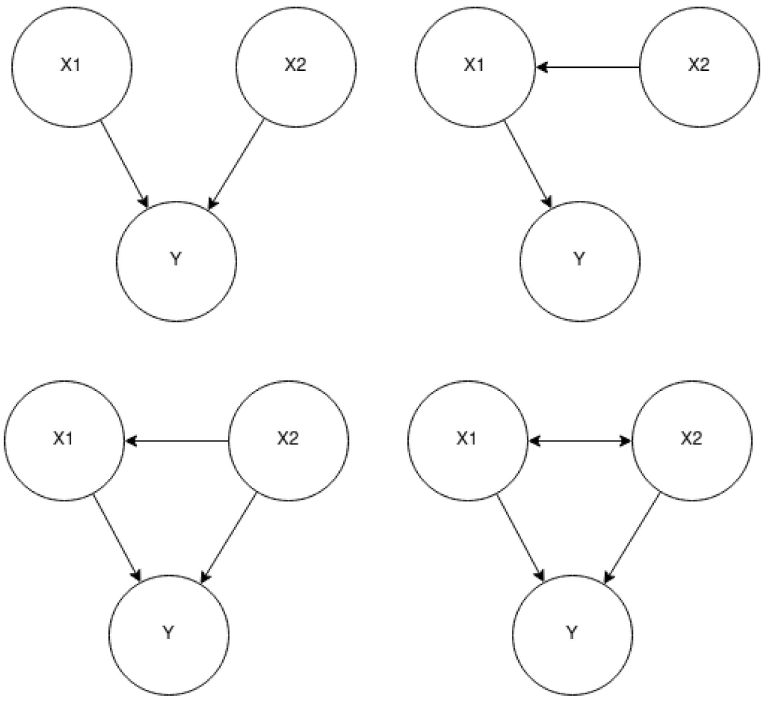

Figure 3.1: Different causal models.

Chapter 4

Methods

This chapter explores in greater detail the individual aspects of our experiment. The aim of the experiment is to compare the SHAP output of two different model structures. First, models are trained with intrinsic explanation, explained in 2.4.3. Next, the intrinsic explanation is removed and each model is retrained. This process is done with CNNs LSTMs and transformers. Finally, the SHAP pot-hoc explanations for the two groups of models (with and without intrinsic explanations) are compared.

4.1 Dataset

The ERASER benchmark is a collection of datasets aimed at advancing research on interpretable NLP models [10] covering several NLP tasks – we focus on the movie review dataset for sentiment analysis. section 4.1 is an abbreviated example of one of the movie reviews. The human annotated explanation and the chosen sentiment are both highlighted in yellow. The human annotations are entirely extractive, and as such, pair well with the forms of explanation used in this work. Annotations for a particular movie review consist of a number of contiguous phrases, giving their start and end word.

4.2 Model Construction

To set up the intrinsic explanation models, we reproduced Lei *et al.* [22] in making a two-module network that learns to predict sentiment based on a sub-

In this movie, ...Plots to take over the world. The acting is great! The soundtrack is run-of-the-mill, but the action more than makes up for it

(a) Positive (b) Negative

Figure 4.1: Example movie review with human annotated explanation highlighted.

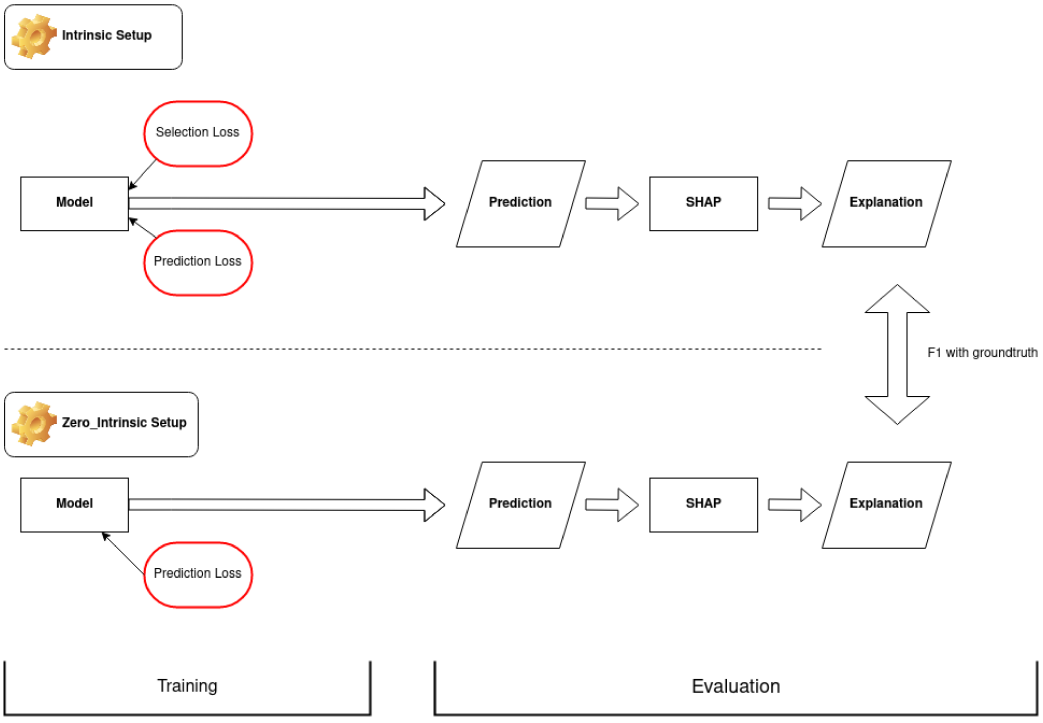

Figure 4.2: Two different structures of models are trained and the SHAP results from each can be compared to groundtruth.

set of the input, chosen by the network itself. This design was chosen because of its simplicity and prior use in NLP. The composition of the generator and encoder need not be any particular type of model; these layers are modular to experiment with CNN, LSTM, and transformer layers. We used SHAP for the post hoc explanation, as it is popular and did not require any internal model parameters.

To compare intrinsic and post hoc explanations, a version of the model without intrinsic explanation is required. To achieve this, we set the loss of the generator to zero to allow the zero_intrinsic model to retain the degrees of freedom that its intrinsic counterpart had.

Models were compared using the experimental design shown in fig. 4.2. Both intrinsic and zero_intrinsic models underwent the same training process. They are then both evaluated with SHAP, and it is these SHAP results that are used to determine model agnosticism.

4.3 Pipeline

To guarantee consistency between trials, a pipeline was used to train, select, and modify individual models. The subsequent subsections illustrate the steps of the pipeline. The ability for SHAP to produce explanations was then compared between counterpart models (e.g. CNNs with and without intrinsic explanation).

4.3.1 Hyperparameter Selection

The training was done with tensorflow in a wrapper of WandB, a service that allows the specification of a space of hyperparameters to search, and it will iterate through the possible combinations based on the given algorithm. If a grid search algorithm is chosen for WandB, then all possible combinations of hyperparameters are tested. In this work, a naive bayes algorithm was chosen, meaning WandB selects hyperparameters from distributions it is provided. These distributions were chosen from earlier prototypes of the models, by time constraints, or by exhaustive selection. As WandB iterates through more runs, it evaluates the performance of models as a function of the hyperparameters chosen; it uses these findings to better predict hyperparameters for future runs.

4.3.2 Preprocessing

An embedding layer in every model converts the language input into vector embeddings. That embedding layer was a hyperparameter that varied by type and size. The embedding was either a fixed Glove [30] embedding or a trainable Keras [8] embedding layer, but in both cases the embedding size ranged from 50 to 300. Based on a model's input size, the training and testing set had to be truncated to the appropriated dimension and the explanation ground

Hyperparameter	Distribution type	Range
Learning rate	log uniform	min:1e-7, max:1e-3
Input size	list	100,200,300,400,500
Drop	uniform	min:0.1 max: 0.5
Embedding size	list	50,100,200,300
Selection Lambda	log uniform	min:0.01, max:1.0
Continuity Lambda	log uniform	min:0.01, max:1.0
CNN filters	list	4,8,16,32,64,128
Transformer heads	list	2,4,8
Transformer dense feedforward	list	32, 64,128
Transformer layers	list	2,4,8
LSTM hidden	list	20,50,100,200

Table 4.1: Distributions of hyperparameters

truths to match.

4.3.3 Model Selection

Once a run of models are trained, they are filtered based on their validation set accuracy in the sentiment analysis task so that the top performing models can be selected for downstream testing. For each architecture, 400 models were trained. The top 40 models evaluated by accuracy in the sentiment analysis task are used for the experiment.

In fig. 4.3 the process is shown where five models begin training as intrinsic models. From the top-k selected models, we derive counterpart models that are identical to the original ones except for the fact that intrinsic explanation regularization is removed. These two intrinsic models are then retrained with their intrinsic loss functions set to zero (shown in purple). The models trained with intrinsic explanation are called "intrinsic", while the models where the intrinsic explanation was stripped away are called "zero_intrinsic".

4.3.4 Model Modification

These selected models are modified so that their intrinsic explanation regulation is removed, and are retrained as the "zero_intrinsic" counterpart. This was done to retain the most similarity between counterpart models; had the intrinsic module been removed, the counterpart model would have had con-

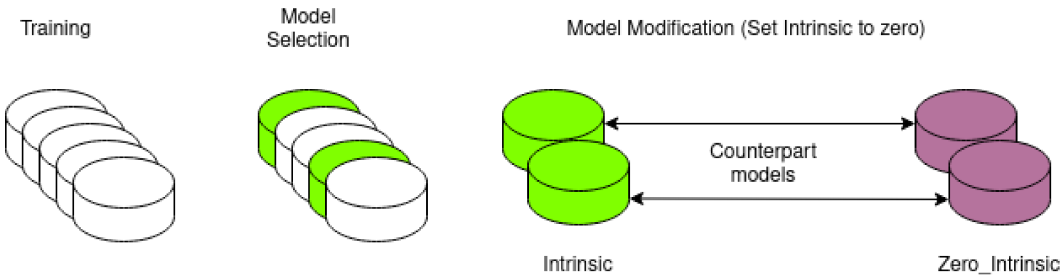

Figure 4.3: Pipeline of model training. It is illustrated here with only 5 models (instead of 400) and the top 2 models chosen (instead of 40).

siderably fewer learnable parameters.

4.3.5 SHAP

Once all models are trained, we use SHAP on each model over the ERASER test set. Note that SHAP returns a weight for each token in the input, indicating how much that token contributed to the prediction. As the human annotations in the benchmark only indicate which tokens explain the prediction, we need to use a cut-off threshold on the SHAP weight to decide whether or not to include the token in the explanation generated by SHAP. We use the following threshold: 1%, 2%, 5%, 10%, 20%, and 50%. With the 1% threshold, we would keep up to 100 tokens that SHAP determined as contributing to the explanation. With the 20% threshold, only up to five tokens would be chosen. once we have SHAP's explanation, we can use the ERASER test set annotations to measure F1 and top_k accuracy.

4.3.6 Evaluation

The models are evaluated using cross-entropy between their ouput and the labels of the ERASER test set. Each model architecture (CNN, LSTM, transformer) had the top 40 models used for statistical analysis.

The Kendall Tau metric [19] provides a value for how similar two rankings are and does so by looking at pairs of values from each ranking. Pairs of values from one ranking that are found in the same order in the other ranking are called concordant pairs, while pairs found in the opposite order are called

Items	Rank1	Rank2	Concordant	Discordant
Matrix	1	1	3	0
Princess Bride	2	2	2	0
Gattaca	3	4	0	1
Airplane	4	3		

Table 4.2: Two rankings of the same four movies, illustrating how the Kendall Tau metric is calculated. It is a metric that gives more weight to the higher ranked elements.

discordant pairs. The Kendall Tau metric is defined as:

$$\tau = (P - Q)/(P + Q)$$

where P is the number of concordant pairs, Q is the number of discordant pairs. The Kendall Tau test value ranged from (-1,1) where -1 signifies that the two lists are in reverse order, 1 for identical order, and 0 means no correlation whatsoever. The Kendall Tau metric was calculated for counterpart models and every combination of non-counterpart models. With this distribution, a percentile can be calculated for where the counterpart models fall in comparison to all other non-counterpart models. If a model is, on average, more similar to its counterpart model than to other models, then it will score very highly in this percentile metric. If counterpart models are completely independent, one would expect a middling percentile. A low percentile would indicate that a random model behaves more similarly than the counterpart model. table 4.2 shows two possible rankings for the same set of movies. Take the sum of the concordant column as P, the sum of the discordant column as Q, and enter them into the formula.

Paired T-tests [36] were used to compare counterpart models. This test is used to compare means of two groups when each observation in one group can be paired with an observation from the second group. The paired t-test gives a notion of whether the treatment, in this case the removal of intrinsic explanation, had an effect beyond random pertubations that may exist in the data.

Chapter 5

Results

We set out to discover whether the underlying structure of a model impacted the explanations generated by SHAP. A number of tests and comparisons were used to answer our research question. SHAP returns a value for each token of the input, and these values can be organized into a ranking of which words contributed most to the prediction by the blackbox model. Firstly, Kendall rank correlation coefficient was used to compare an intrinsic model with its counterpart where the intrinsic explanation was removed. This was done to give confidence that the modified "zero_intrinsic" models are still acting as small variations on the same model, rather than two different models. Secondly, the F1 score and the top k accuracy of the models were compared in paired t-tests. These t-tests give a metric of statistical significance for whether removing the intrinsic cost functions affected SHAP's performance. Finally, an analysis of the different types of models and hyperparameters was performed.

5.1 Ranking

The Kendall Tau metric is used to compare rankings of all pairs of models, to see how similar counterpart models are. It was found that, a model behaves more like its counterpart model, on average, than to any other model.

The results from Kendall Tau comparisons are presented in table 5.1. This shows that on average a model was among the most similar to its counterpart model than any other model.

Model	Percentile Mean
Transformer	85.47
LSTM	79.11
CNN	75.71

Table 5.1: Where the counterpart models lie in the distribution of the Kendall Tau metric between all pairs of models. This shows that counterpart models were generally much more similar to each other than to any other pairing of models. To be the 100^{th} percentile means that the counterpart models were the most similar models. It is a percentile mean because this percentile is averaged across all models.

5.2 Comparisons

To calculate the importance of intrinsic explanations in a model, pairwise T-Tests were conducted between the counterpart models. In table 5.2, the performance of the different styles of model, both with("-intrinsic") and without intrinsic ("-zero") explanation, are shown. Two models' performance from the dataset are also included at the top.

Model	Average Model Accuracy	Average F1 Score
Bert-to-Bert pipeline	0.860	0.145
(Lehman et al., 2019) pipeline	0.750	0.139
Transformer-intrinsic	0.759	0.148
Transformer-zero	0.755	0.143
LSTM-intrinsic	0.718	0.176
LSTM-zero	0.718	0.173
CNN-zero	0.770	0.140
CNN-intrinsic	0.772	0.131

Table 5.2: Evaluation metrics compared to ERASER dataset numbers

To ascertain whether differences in the counterpart models had statistical significance, p-values for table 5.2 are given in table 5.3. The CNN architecture shows statistical significance in both F1 and top_k across all thresholds, while notably having no statistical significance between model accuracies. This indicates that for the CNN architecture, SHAP is influenced by model structure. To see that, note that if SHAP was unbiased towards the underlying structure then there would not be such a strong statistical significance across such a large paired test. We cannot attribute this statistical significance to differences

in accuracy, as there was no statistical significance found in that comparison. Neither the transformer nor the LSTM showed statistical significance in either model accuracy or explanation metrics.

In table 5.4, side by side comparisons are shown for the intrinsic and zero_intrinsic structures across all three architectures. CNNs show the zero_intrinsic models as the superior model in every category except model accuracy. This was reflected in table 5.3 where statistical significance in this difference was found. Despite the lack of statistical significance in evaluation metrics for transformer, a trend of zero_intrinsic outperformance seems to be present. The LSTM architecture displays a much more balanced result.

T-test performed	CNN	Transformer	LSTM
Model accuracy	0.67	0.46	0.95
F1 @ 0.01	0.011	0.092	0.163
F1 @ 0.02	0.017	0.265	0.859
F1 @ 0.05	0.017	0.389	0.791
F1 @ 0.10	0.013	0.406	0.627
F1 @ 0.20	0.013	0.459	0.396
F1 @ 0.50	0.015	0.435	0.449
Top_k @ 1	0.001	0.651	0.331
Top_k @ 5	0.000	0.482	0.308
Top_k @ 10	0.000	0.470	0.280
Top_k @ 20	0.000	0.592	0.807

Table 5.3: P-values from pairwise T-tests when comparing the intrinsic model set with the zero_intrinsic model set. P-values indicating statistical significance are highlighted in blue. Statistical significance for the F1 score and top_k indicates whether SHAP's explanations were affected by the removal of intrinsic explanation. There was no statistical significance in model accuracy in any architecture.

Metrics	CNN-int	CNN-zero	Tra-int	Tra-zero	LSTM-int	LSTM-zero
Model accuracy	0.772	0.770	0.759	0.755	0.718	0.718
F1 @ 0.01	0.131	0.140	0.143	0.148	0.176	0.173
F1 @ 0.02	0.122	0.136	0.136	0.141	0.169	0.170
F1 @ 0.05	0.116	0.134	0.133	0.138	0.144	0.145
F1 @ 0.10	0.116	0.135	0.134	0.138	0.138	0.135
F1 @ 0.20	0.116	0.135	0.135	0.139	0.137	0.132
F1 @ 0.50	0.117	0.135	0.136	0.140	0.137	0.133
Top_k @ 1	0.100	0.114	0.166	0.169	0.142	0.150
Top_k @ 5	0.098	0.110	0.144	0.147	0.145	0.149
Top_k @ 10	0.096	0.105	0.135	0.133	0.133	0.136
Top_k @ 20	0.093	0.099	0.122	0.121	0.123	0.124

Table 5.4: Metrics for each architecture-structure pairing. Transformer is abbreviated to "tra". Colouring indicates the better performing model between the two different model structures for each architecture.

The accuracy of the top k results from SHAP being in the ground truth are illustrated in figs. 5.1, 5.3 and 5.5. The F1 score for varying levels of threshold for which to include each word for the extractive explanation are shown in figs. 5.2, 5.4 and 5.6.

The results in figs. 5.1 to 5.6 show three different subsets of the data: "all" which is the entire dataset, "correct" which are only the rows the model answered the sentiment analysis task correctly, and "confident" which are the rows the model was confident. Confidence was defined as a prediction made at or higher than 60%. The results from the different thresholds support the notion that the generated explanations are explaining the model, because the "correct" subset generally performs the best. Since the explanations are generated to make the prediction, an error in prediction could be the result of a poor explanation.

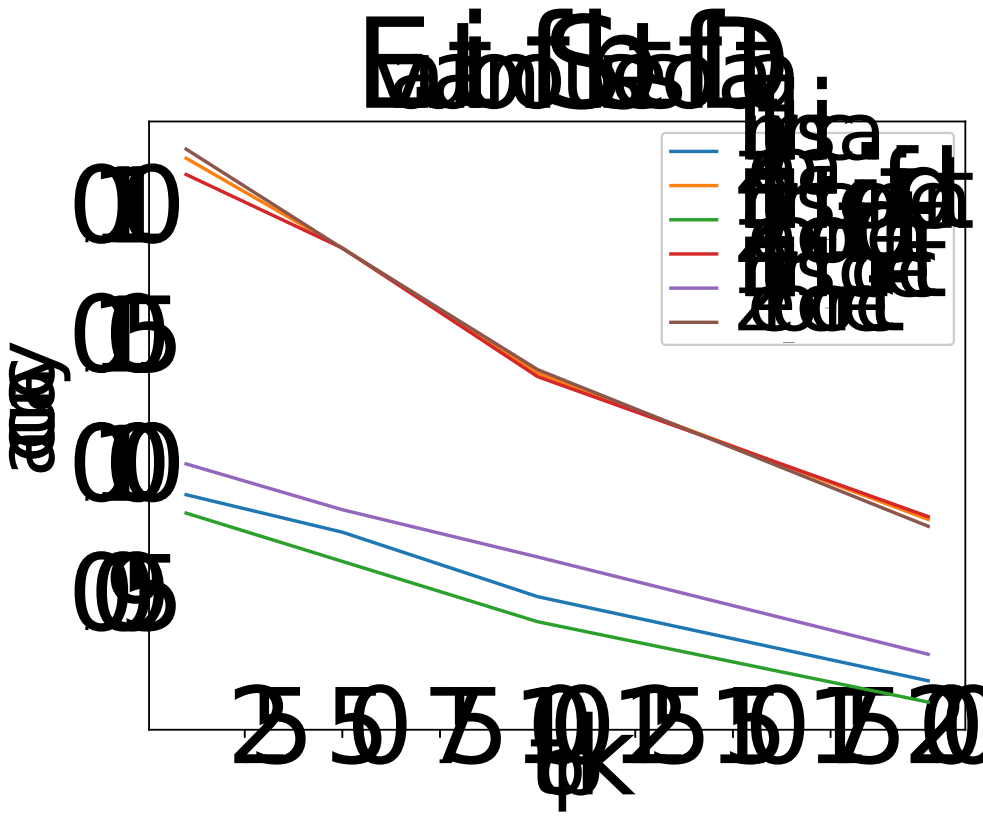

Figure 5.1: Measuring top k accuracy for different values of k for CNN models. This graph shows a clear gap between intrinsic and zero_intrinsic models across all subsets of data. The top k accuracy does not seem to be much affected by which subset of data.

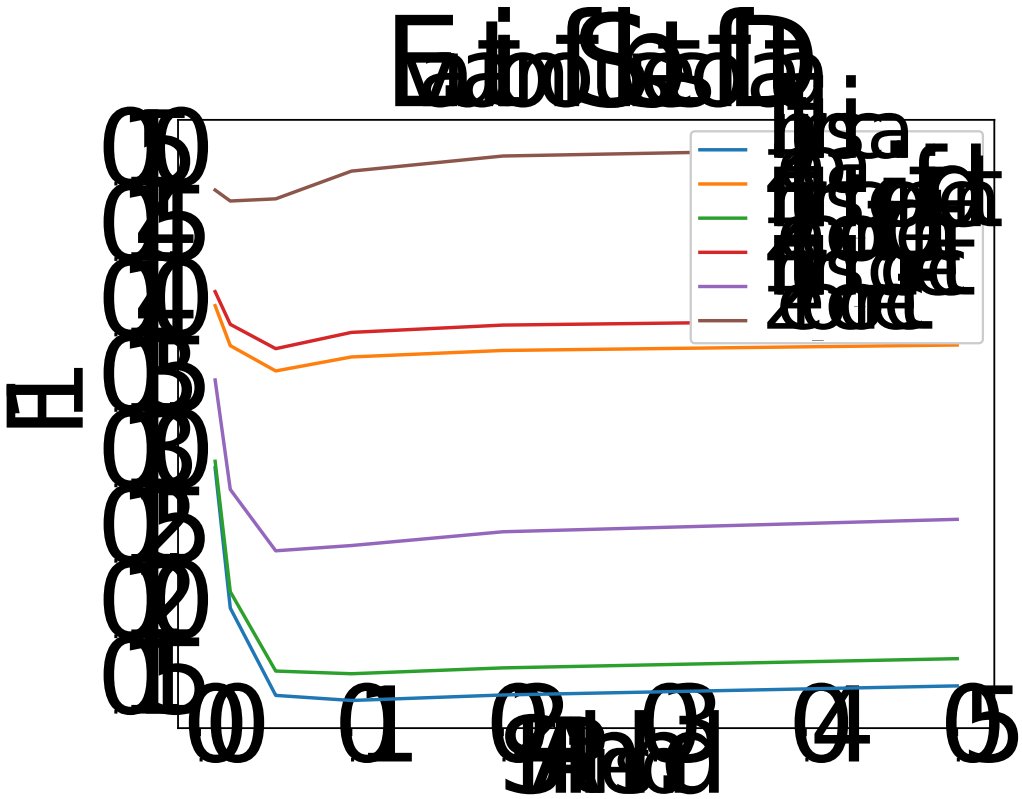

Figure 5.2: Measuring F1 score across varying thresholds for converting SHAP values to explanations for CNN models. Choice of subset has a noticable effect on performance, however all zero_intrinsic treatment runs outperform intrinsic treatment runs. In both model treatments, explanation F1 score was improved when only considering when the model was confident, the best results were when considering only the instances where it was correct.

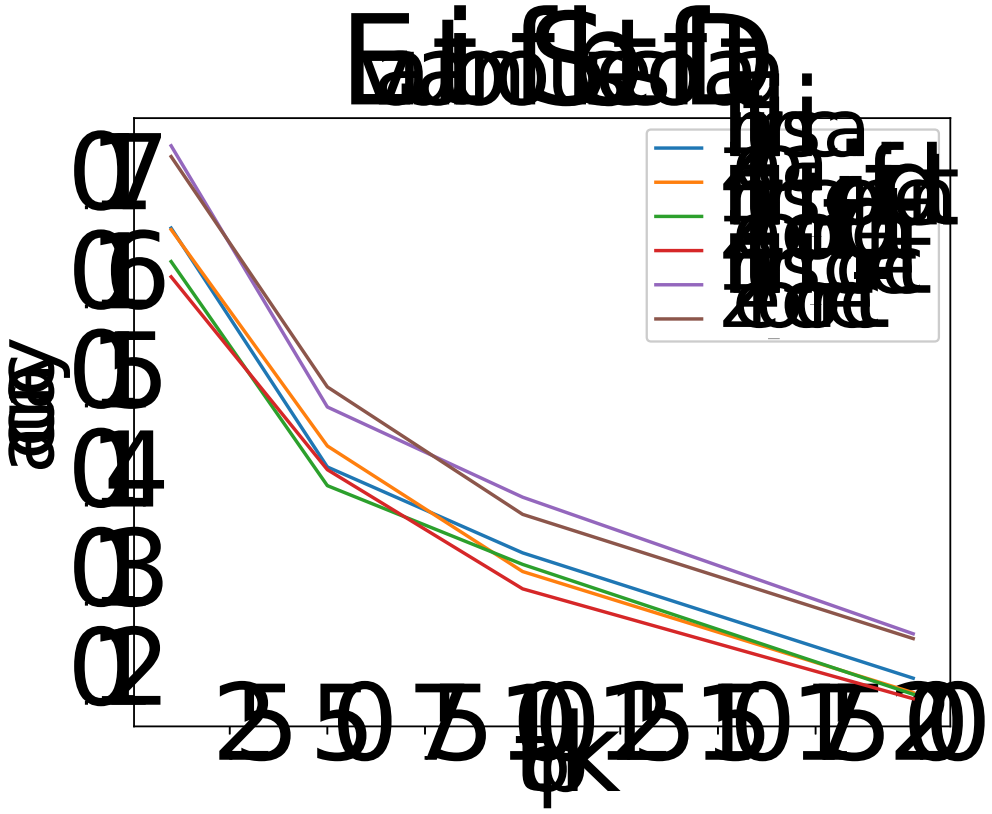

Figure 5.3: Measuring top k accuracy for different values of k for transformer models. This graph shows there is no clear difference between model treatment. Depending on data subset, intrinsic or zero_intrinsic may be best. Across most values of k, the accuracy is improved slightly when considering only the confident subset, and improved more when considering the correct subset.

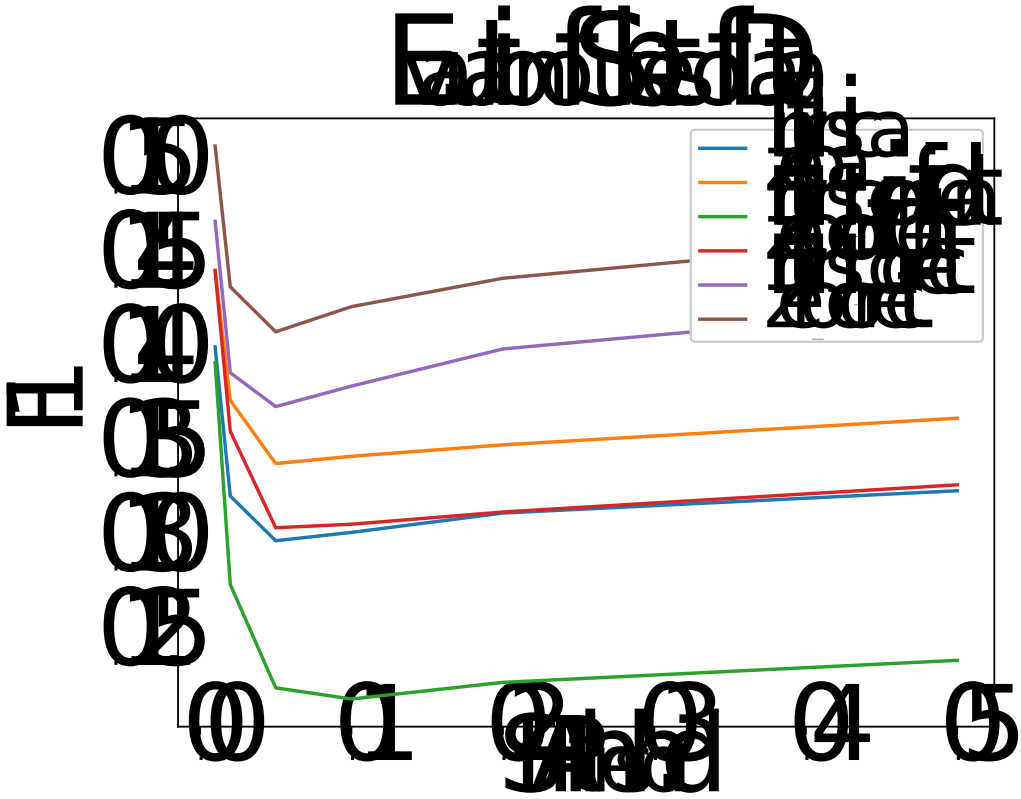

Figure 5.4: Measuring F1 score across varying thresholds for converting SHAP values to explanations for transformer models. Intrinsic and zero_intrinsic treatments are grouped by subset, with the correct subset outperforming and the confident subset underperforming the entire dataset.

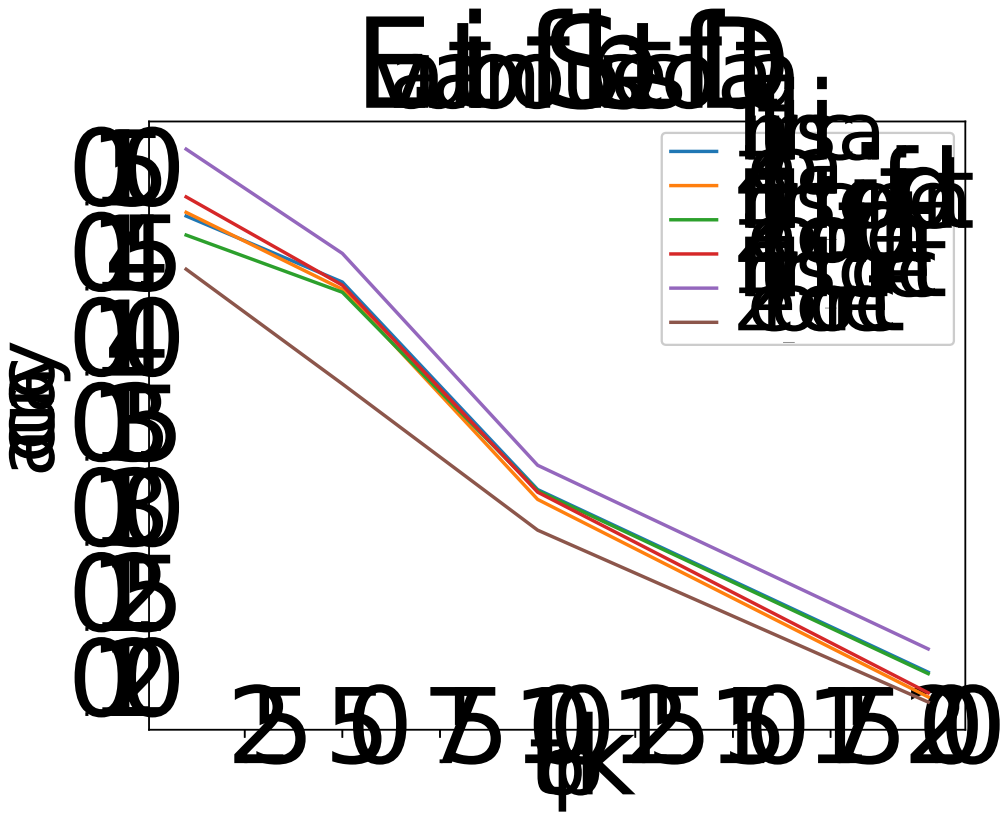

Figure 5.5: Measuring top k accuracy for different values of k for LSTM models. All lines, except for the correct subset, are tightly intertwined. The intrinsic treatment outperforms the zero_intrinsic treatment for this correct subset.

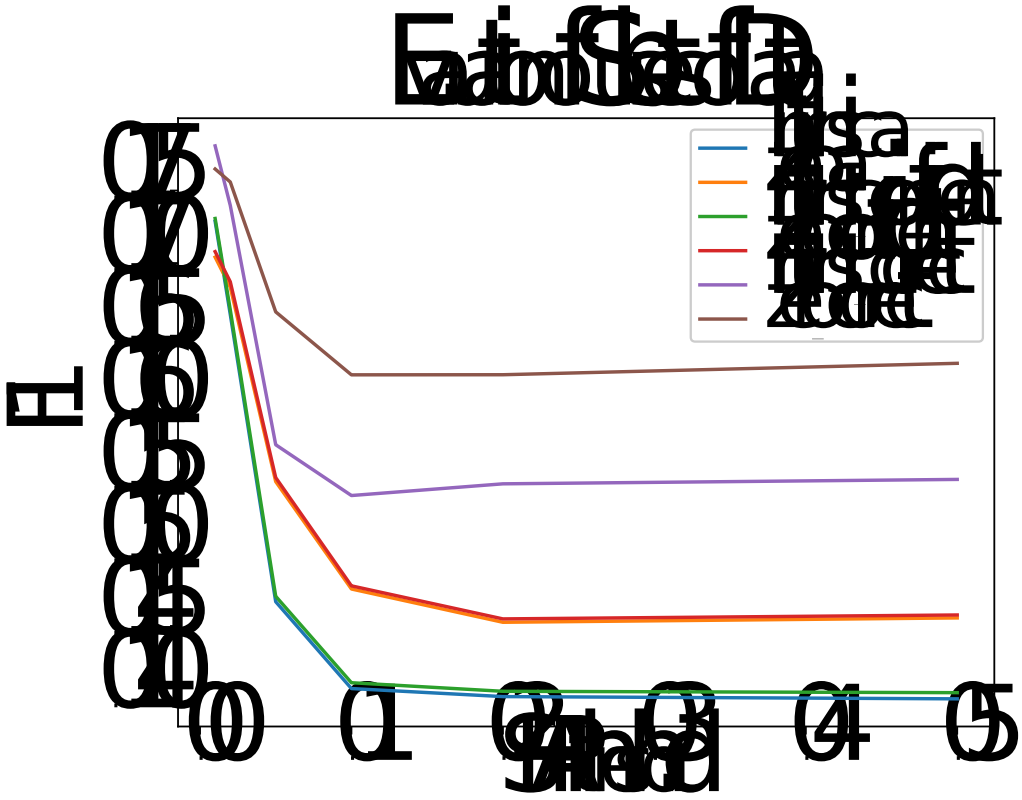

Figure 5.6: Measuring F1 score across varying thresholds for converting SHAP values to explanations for LSTM models. The confident subsection was almost identical to the entire dataset for both the intrinsic and zero_intrinsic treatments. The zero_intrinsic treatment outperforms the intrinsic treatment for the correct subset.

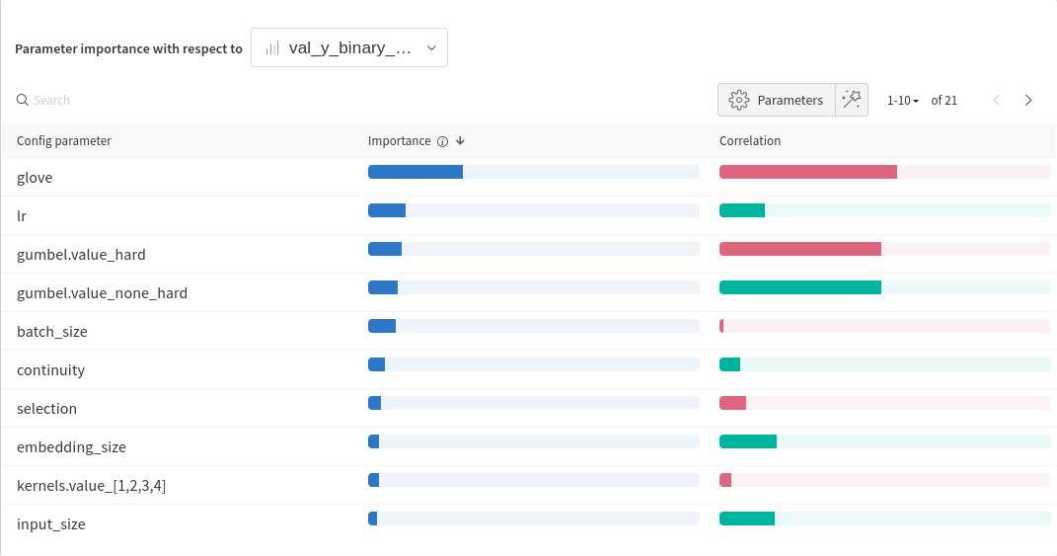

Figure 5.7: Hyperparameter importance for the CNN models while training. Glove, as opposed to a Keras embedding, is a hindrance to predictive accuracy of models in training. In the "Correlation" column, red indicates a negative number, green indicates a positive number, and the size of the bar indicates magnitude. In the "Importance" column, it is only the magnitude to be considered.

5.3 Hyperparameters

Model specific hyperparameter correlation and importance were calculated to facilitate a better use of training time. These metrics are calculated in WandB and an example is shown in fig. 5.7. This figure offers two metrics: correlation and importance. Correlation is a measure of linear correlation between a hyperparameter and some model metric. Importance is the result of training a random forest using hyperparameters as input to predict the model metric.

A linear model was also fit to see what variance could be explained for both the predicting ability and the F1 score for each architecture in tables 5.5 to 5.10.

Glove embeddings had different effects on the training and the evaluaion of SHAP. For both CNN and transformer architectures, glove embeddings were a negative influence on the prediction results, glove embeddings actually gave SHAP explanations a higher F1 score. This effect was statistically significant in both prediction and F1 score. This may be because glove embeddings were

Parameter	Coef	Std. Err.	T-value	P-value
Intercept	0.7351	0.027	27.269	0.000
batch-size	-0.0001	0.000	-0.607	0.546
selection	-0.0636	0.047	-1.345	0.183
continuity	0.0030	0.014	0.209	0.835
drop	0.0484	0.027	1.787	0.078
embedding-size	7.801e-05	5.31e-05	1.469	0.146
glove	-0.0504	0.007	-7.444	0.000
input-size	7.474e-05	2.67e-05	2.800	0.007

Table 5.5: Linear regression results for the CNN models. Highlighting indicates a p-value < 0.05. Resulting coefficients from fitting the equation: prediction accuracy= batch-size + selection +continuity + drop + embedding-size + glove + input-size
This shows glove's statistical significance in determining a model's prediction score for the resulting explanation from SHAP. The R^2 value was .504

Parameter	Coef	Std. Err.	T-value	P-value
Intercept	0.1433	0.021	6.912	0.000
batch-size	6.276e-05	0.000	0.374	0.709
selection	-0.0428	0.036	-1.179	0.243
continuity	-0.0039	0.011	-0.358	0.722
drop	0.0155	0.021	0.746	0.458
embedding-size	5.738e-06	4.08e-05	0.140	0.889
glove	0.0255	0.005	4.897	0.000
input-size	-5.171e-05	2.05e-05	-2.519	0.014

Table 5.6: Linear regression results for the CNN models. Highlighting indicates a p-value < 0.05. Resulting coefficients from fitting the equation: F1 = batch-size + selection +continuity + drop + embedding-size + glove + input-size
This shows glove's statistical significance in determining a model's F1 score for the resulting explanation from SHAP. The R^2 value was .382

trained to differentiate words, while the Keras embeddings are only focused on maximizing predictive score. Because Keras embeddings are a trained layer, their loss function will be in line with prediction accuracy instead of explanation F1 score. Glove embeddings, on the other hand, have been previously trained to give some notion of word similarity based on co-occurence. The information and relations learned through the training of Glove embeddings seems to produce better explanation scores.

Parameter	Coef	Std. Err.	T-value	P-value
Intercept	0.7506	0.023	32.212	0.000
batch-size	-0.0002	0.000	-0.601	0.551
selection	-0.0592	0.107	-0.552	0.584
continuity	0.0207	0.014	1.517	0.136
drop	0.0009	0.044	0.020	0.984
embedding-size	0.0002	5.08e-05	3.049	0.004
glove	-0.0481	0.011	-4.385	0.000
input-size	8.009e-05	4.04e-05	1.981	0.053

Table 5.7: Linear regression results for the Transformer models. Highlighting indicates a p-value < 0.05. Resulting coefficients from fitting the equation: prediction accuracy= batch-size + selection +continuity + drop + embedding-size + glove + input-size
This shows glove's statistical significance in determining a model's prediction score for the resulting explanation from SHAP. The R^2 value was .335 .

Parameter	Coef	Std. Err.	T-value	P-value
Intercept	0.1746	0.017	10.107	0.000
batch-size	-0.0004	0.000	-1.841	0.072
selection	0.0139	0.080	0.174	0.862
continuity	0.0073	0.010	0.718	0.476
drop	0.0272	0.033	0.832	0.409
embedding-size	6.103e-06	3.77e-05	0.162	0.872
glove	0.0187	0.008	2.302	0.026
input-size	-0.0001	3e-05	-3.934	0.000

Table 5.8: Linear regression results for the Transformer models. Highlighting indicates a p-value < 0.05. Resulting coefficients from fitting the equation: F1 = batch-size + selection +continuity + drop + embedding-size + glove + input-size
This shows glove's statistical significance in determining a model's F1 score for the resulting explanation from SHAP. The R^2 value was .418.

Parameter	Coef	Std. Err.	T-value	P-value
Intercept	0.6617	0.024	27.251	0.000
batch-size	1.018e-05	6.34e-05	0.161	0.874
selection	0.0306	0.032	0.965	0.345
continuity	0.0128	0.016	0.787	0.440
drop	0.0430	0.024	1.788	0.088
embedding-size	2.787e-05	8.2e-05	0.340	0.737
glove	0.0193	0.012	1.560	0.133
input-size	0.0002	7.91e-05	2.257	0.034

Table 5.9: Linear regression results for the LSTM models. Highlighting indicates a p-value < 0.05. Resulting coefficients from fitting the equation: prediction accuracy= batch-size + selection +continuity + drop + embedding-size + glove + input-size
This indicates only statistical significance for input-size, however at a very low value. The R^2 value was 0.491.

Parameter	Coef	Std. Err.	T-value	P-value
Intercept	0.1831	0.015	11.870	0.000
batch-size	4.631e-05	4.03e-05	1.150	0.263
selection	0.0025	0.020	0.122	0.904
continuity	-0.0110	0.010	-1.069	0.296
drop	-0.0051	0.015	-0.332	0.743
embedding-size	5.975e-05	5.21e-05	1.147	0.264
glove	-0.0060	0.008	-0.765	0.452
input-size	-0.0002	5.02e-05	-3.195	0.004

Table 5.10: Linear regression results for the LSTM models. Highlighting indicates a p-value < 0.05. Resulting coefficients from fitting the equation: F1 = batch-size + selection +continuity + drop + embedding-size + glove + input-size
This indicates only statistical significance for input-size, however at a very low value. The R^2 value was 0.479.

Chapter 6

Conclusion

In this work, we investigated whether model structure could affect the quality of SHAP explanations in a sentiment analysis task. To do that investigation, we compared multiple counterpart models from three different architectures (CNN, LSTM, Transformer) with and without intrinsic explanations.

When comparing F1 scores and accuracy at top k between these counterpart pairs of structures, the only model architecture that had a statistically significant difference was the CNN architecture. Notably, the difference in predictive accuracy between the models with intrinsic explanation and those without, was not statistically significant. These two observations indicate that model structure has a role in determining SHAP efficacy. If SHAP was not affected by model structure, then we would have expected to see either a change in F1 score correlating with prediction accuracy, or no change in F1 score between model structures.

6.1 Limitations

The sentiment analysis dataset from Eraser has only 2000 samples. With such a small dataset, we can't assume that the statistical significance will hold for all sentiment analysis tasks or even a larger movie review dataset.

6.2 Discussion

The question remains for why the effect measured in the CNN models may have happened. To attempt to answer this, refer to how SHAP calculates feature importance and how the model intrinsic explanation was constructed. SHAP perturbs the input iteratively, assigning more importance to features that are more often important in model prediction. The intrinsic explanation has a generator layer to mask input tokens that is only ever trained on complete inputs. This means that there is no notion of whether the generator (the first module of the model, shown in fig. 2.7 works for strange input combinations that SHAP may provide. While the generator may provide adequate explanations for regular language, an input that has already had a number of tokens masked may be out of scope for the generator. This may result in the generator masking tokens that would not have been masked if given the full input, which could negatively impact SHAP's ability to determine which tokens contributed the most to the blackbox output.

However, the reason for the difference in F1 may not have been solely due to the intrinsic explanation, but could have been the explanation treatment in conjunction with model architecture. Statistical significance was not found across the board in the comparison of explanation treatment types. It was only found in the CNN models. A key element of the CNN design is a kernel that convolves the input. This kernel has limited context and may not be able to cope with having masked inputs.

The significance of these results is that they show there may be a way to increase a model's explanation score, without significantly changing model training. Whether this increase in model explanation score correlates to more explainable models or whether it is simply a flaw in SHAP's calculations is not answered with these results. If the increase in model explanation score does not provide an explanation that a human finds more helpful, then this may be a way to game the SHAP value.

6.3 Future Work

A large issue for SHAP is the interdependency of features. Following Chen and Jordan [7], a parse tree could be input to SHAP to use instead of the default partition scheme. The default partition scheme illustrated in fig. 2.5 and fig. 2.6, does not use any linguistic knowledge to make the partitions. This could alleviate a major hurdle for dealing with natural language at the same time as injecting more domain information into the process.

If SHAP or similar libraries maintain their relevance in XAI, developing a dataset geared towards their style could be beneficial. SHAP provides scores to every token, which can be positive or negative for the particular class prediction. The Eraser datasets provide only explanations that contribute to the actual prediction, rather than also explanations for what hindered the prediction. The annotation process could instead be structured so that for each prediction class, annotators find phrases that contribute to that class.

Since glove embeddings accounted for such a large portion of the variance in the results, it seems a good avenue to explore whether other embeddings can further improve SHAP. In particular, an approach of interpretable embeddings like in Panigrahi *et al.* [28] could yield interesting results.

References

[1] J. Alammar. "The illustrated transformer." (), [Online]. Available: https://jalammar.github.io/illustrated-transformer/ (visited on 10/06/2022).

[2] J. Bastings and K. Filippova, "The elephant in the interpretability room: Why use attention as explanation when we have saliency methods?" In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2020, Online, November 2020*, A. Alishahi, Y. Belinkov, G. Chrupala, D. Hupkes, Y. Pinter, and H. Sajjad, Eds., Association for Computational Linguistics, 2020, pp. 149–155. DOI: 10.18653/v1/2020.blackboxnlp-1.14. [Online]. Available: https://doi.org/10.18653/v1/2020.blackboxnlp-1.14.

[3] M. Bellucci, N. Delestre, N. Malandain, and C. Zanni-Merk, "Towards a terminology for a fully contextualized XAI," in *Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 25th International Conference KES-2021, Virtual Event / Szczecin, Poland, 8-10 September 2021*, J. Watróbski, W. Salabun, C. Toro, C. Zanni-Merk, R. J. Howlett, and L. C. Jain, Eds., ser. Procedia Computer Science, vol. 192, Elsevier, 2021, pp. 241–250. DOI: 10.1016/j.procs.2021.08.025. [Online]. Available: https://doi.org/10.1016/j.procs.2021.08.025.

[4] A. Bokulich, "How scientific models can explain," *Synthese*, vol. 180, pp. 33–45, 2011.

[5] T. B. Brown, B. Mann, N. Ryder, *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

[6] D. Cer, Y. Yang, S. Kong, *et al.*, "Universal sentence encoder," *CoRR*, vol. abs/1803.11175, 2018. arXiv: 1803.11175. [Online]. Available: http://arxiv.org/abs/1803.11175.

[7] J. Chen and M. I. Jordan, "Ls-tree: Model interpretation when the data are linguistic," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, AAAI Press, 2020, pp. 3454–3461. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/5749.

[8] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.

[9] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds., Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: 10.18653/v1/n19-1423. [Online]. Available: https://doi.org/10.18653/v1/n19-1423.

[10] J. DeYoung, S. Jain, N. F. Rajani, *et al.*, "ERASER: A benchmark to evaluate rationalized NLP models," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, Eds., Association for Computational Linguistics, 2020, pp. 4443–4458. DOI: 10.18653/v1/2020.acl-main.408. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.408.

[11] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision-making and a "right to explanation"," *AI magazine*, vol. 38, no. 3, pp. 50–57, 2017.

[12] C. Grimsley, E. Mayfield, and J. R. S. Bursten, "Why attention is not explanation: Surgical intervention and causal reasoning about neural models," in *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, N. Calzolari, F. Béchet, P. Blache, *et al.*, Eds., European Language Resources Association, 2020, pp. 1780–1790. [Online]. Available: https://aclanthology.org/2020.lrec-1.220/.

[13] T. Heskes, E. Sijben, I. G. Bucur, and T. Claassen, "Causal shapley values: Exploiting causal knowledge to explain individual predictions of complex models," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/32e54441e6382a7fbacbbbaf3c450059-Abstract.html.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *CoRR*, vol. abs/1508.01991, 2015. arXiv: 1508.01991. [Online]. Available: http://arxiv.org/abs/1508.01991.

[16] H. Ij, "Statistics versus machine learning," *Nat Methods*, vol. 15, no. 4, p. 233, 2018.

[17] A. Jacovi and Y. Goldberg, "Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness?" In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, Eds., Association for Computational Linguistics, 2020, pp. 4198–4205. DOI: 10.18653/v1/2020.acl-main.386. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.386.

[18] S. Jain and B. C. Wallace, "Attention is not explanation," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds., Association for Computational Linguistics, 2019, pp. 3543–3556. DOI: 10.18653/v1/n19-1357. [Online]. Available: https://doi.org/10.18653/v1/n19-1357.

[19] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1/2, pp. 81–93, 1938.

[20] D. Klein and C. D. Manning, "Conditional structure versus conditional estimation in nlp models," in *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, 2002, pp. 9–16.

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. DOI: 10.1109/5.726791. [Online]. Available: https://doi.org/10.1109/5.726791.

[22] T. Lei, R. Barzilay, and T. S. Jaakkola, "Rationalizing neural predictions," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, J. Su, X. Carreras, and K. Duh, Eds., The Association for Computational Linguistics, 2016, pp. 107–117. DOI: 10.18653/v1/d16-1011. [Online]. Available: https://doi.org/10.18653/v1/d16-1011.

[23] S. M. Lundberg and S. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, *et al.*, Eds., 2017, pp. 4765–4774. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html.

[24] S. M. Lundberg and S. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, *et al.*, Eds., 2017, pp. 4765–4774. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html.

[25] A. Madsen, S. Reddy, and S. Chandar, "Post-hoc interpretability for neural NLP: A survey," *CoRR*, vol. abs/2108.04840, 2021. arXiv: 2108.04840. [Online]. Available: https://arxiv.org/abs/2108.04840.

[26] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2013. [Online]. Available: http://arxiv.org/abs/1301.3781.

[27] C. Molnar, G. Casalicchio, and B. Bischl, "Quantifying interpretability of arbitrary machine learning models through functional decomposition," *CoRR*, vol. abs/1904.03867, 2019. arXiv: 1904.03867. [Online]. Available: http://arxiv.org/abs/1904.03867.

[28] A. Panigrahi, H. V. Simhadri, and C. Bhattacharyya, "Word2sense: Sparse interpretable word embeddings," in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, A. Korhonen, D. R. Traum, and L. Màrquez, Eds., Association for Computational Linguistics, 2019, pp. 5692–5705. DOI: 10.18653/v1/p19-1570. [Online]. Available: https://doi.org/10.18653/v1/p19-1570.

[29] J. Pearl, "The do-calculus revisited," in *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, August 14-18, 2012*, N. de Freitas and K. P. Murphy, Eds., AUAI Press, 2012, pp. 3–11. [Online]. Available: https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1%5C&smnu=2%5C&article%5C_id=2330%5C&proceeding%5C_id=28.

[30] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October*

25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, A. Moschitti, B. Pang, and W. Daelemans, Eds., ACL, 2014, pp. 1532–1543. DOI: 10.3115/v1/d14-1162. [Online]. Available: https://doi.org/10.3115/v1/d14-1162.

[31] 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, San Francisco, CA, USA, August 13-17, 2016*, B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, Eds., ACM, 2016, pp. 1135–1144. DOI: 10.1145/2939672.2939778. [Online]. Available: https://doi.org/10.1145/2939672.2939778.

[32] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, S. A. McIlraith and K. Q. Weinberger, Eds., AAAI Press, 2018, pp. 1527–1535. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16982.

[33] L. S. Shapley, "17. a value for n-person games," in *Contributions to the Theory of Games (AM-28), Volume II*, Princeton University Press, 2016, pp. 307–318.

[34] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro, "Megatron-lm: Training multi-billion parameter language models using model parallelism," *CoRR*, vol. abs/1909.08053, 2019. arXiv: 1909.08053. [Online]. Available: http://arxiv.org/abs/1909.08053.

[35] R. A. Stine, "Sentiment analysis," *Annual review of statistics and its application*, vol. 6, pp. 287–308, 2019.

[36] Student, "The probable error of a mean," *Biometrika*, vol. 6, no. 1, pp. 1–25, 1908.

[37] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 828–841, 2019. DOI: 10.1109/TEVC.2019.2890858. [Online]. Available: https://doi.org/10.1109/TEVC.2019.2890858.

[38] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International conference on machine learning*, PMLR, 2017, pp. 3319–3328.

[39] A. J. Tixier, "Notes on deep learning for NLP," *CoRR*, vol. abs/1808.09772, 2018. arXiv: 1808.09772. [Online]. Available: http://arxiv.org/abs/1808.09772.

[40] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, *et al.*, Eds., 2017, pp. 5998–6008. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[41] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., Association for Computational Linguistics, 2019, pp. 11–20. DOI: 10.18653/v1/D19-1002. [Online]. Available: https://doi.org/10.18653/v1/D19-1002.

[42] J. K. Winkler, C. Fink, F. Toberer, *et al.*, "Association between surgical skin markings in dermoscopic images and diagnostic performance of a deep learning convolutional neural network for melanoma recognition," *JAMA dermatology*, vol. 155, no. 10, pp. 1135–1141, 2019.

[43] Q. Xie, Z. Dai, E. H. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/44feb0096faa8326192570788b38c1d1-Abstract.html.