ENGINEERING
DEPARTMENT OF ELECTRICAL,
COMPUTER, AND SOFTWARE ENGINEERING

2023 Part IV Research Project

Final Report

Automated Generation of Trustworthiness Oracles for Machine Learning Models

Prepared by: Steven Cho (scho518)

Project number: 101
Project partner: Seaton Cousins-Baxter
Supervisor: Valerio Terrangi

Declaration of Originality

This report is my own unaided work and was not copied from anyone or anywhere nor written in collaboration with any other person.

Author: Steven Cho

13th October 2023

Abstract

In this paper, we propose TOWER, a novel use of word embeddings to automatically evaluate the trustworthiness of a model-agnostic classifier based on the outputs of explanatory techniques. This would be the first fully-automated method to measure the trustworthiness of machine learning models. We evaluate this tool through training trustworthy and untrustworthy models via noise-based methods, as well as creating a ground-truth human-labelled explanation trustworthiness dataset to compare it with. We find that, unfortunately, TOWER does not perform as well as hoped. We conclude that further work will be needed to turn this into a fully effective tool to use.

1 Introduction

1.1 Background

We are becoming more and more reliant on machine learning in important parts of society, and with it comes the importance of knowing whether to trust the predictions of the model. While the field of testing the accuracy of models' predictions have been thorough, there has not been as much development for evaluating how much we can trust said predictions.

A large focus of the machine learning space is the maximization of the accuracy of models. However, testing is also done in other areas, such model relevance, efficiency, robustness, fairness, and interpretability [1]. However, much less explored is the space of determining, quantifiably, how far a model can be trusted.

There have been various proposals to measure the trustworthiness of models, but almost all require human involvement in the process [2, 3, 4]. There has been seemingly no attempt

at creating an automated method to measure the trustworthiness of the models.

Thus, we propose TOWER, a novel technique that uses explainability methods and word embeddings to automatically evaluate a metric of trustworthiness of a given classification system.

TOWER is based on explainability techniques. There are many types of types [5], but the relevant category is feature attribution methods – that highlights the features of a model's input that most affect the output. In this case, these would be specific words in a textual input. Then, word embeddings can be used to determine the features' relatedness; that is, how conceptually closely connected the model's justifications of a prediction is to the prediction itself. This method can then be used on a multitude of test cases to determine the overall trustworthiness of the model.

This rests on the assumption that the more related the model's justifications are, the more it would make sense for a human observer, and thus a highest level of trustworthiness. This is an intuitive statement which seems to be supported by others, though with no rigorous backing [6]. For example, say we have two movie review classifiers that classified a review as 'positive'. If the first had the justifications 'it', 'review', 'good', and the second had 'good', 'fantastic', 'amazing', then intuitively one would say the latter is more trustworthy.

Of course, there is the issue that word embeddings themselves are models that could have their own trustworthiness problems [7, 8, 9, 10]. This can be mitigated via the use of ensembles [11].

In addition, TOWER can be expanded to image classifiers as well via the use of another image classifier to convert the identified features into words, which is then fed into the word embeddings as above.

1.2 Objectives

Thus, our objectives are:

- To explore the state-of-the-art in machine learning trustworthiness testing.
- To develop an automated method for quantitatively evaluating the trustworthiness of machine learning models through the novel use of word embeddings on the results from explainability techniques.
- To evaluate its effectiveness on real-world machine learning models.

1.3 Significance

Automating the testing of machine learning models for trustworthiness could have significant benefits, including saving time, reducing costs, and enabling rapid identification of useful models. This approach could also help uncover hidden issues in existing models and provide an opportunity for further research. By eliminating the need for a human component in trustworthiness evaluation, the efficiency at which one could evaluate a model this way would increase by several orders of magnitude. It could also be used within model training to automatically improve a machine learning model.

1.4 Contributions

Our main contributions are as follows:

- TOWER, a method that can automatically evaluate the trustworthiness of any classifier.

- A dataset of explanations created via various types of machine learning models trained on various types and levels of noise.
- A ground-truth dataset of explanations human-labelled as trustworthy or not.

1.5 Scope

This research project will focus on the development of an automated trustworthiness evaluation method for black box models. The method will be done on specifically classifiers, due to the nature of explainability techniques, and will use labelled test cases to do so. It will be text-based, though images could be a possibility. The project will focus on the evaluation of trustworthiness on text classifiers, with brief mention of image classifiers.

2 Related work

Various background work and concepts are discussed here.

2.1 Machine learning testing

A fundamental problem in machine learning is that models are designed to answer questions that have no previous answer, and thus very difficult to test compared to traditional software programs. This is known as the oracle problem [12].

Traditionally, the main metric models are measured by is accuracy. While this has been studied extensively [13], one should not judge a model solely by its accuracy and related metrics, as this does not fully represent the workings of the model.

There are a multitude of ways to test models, as well as a multitude of metrics to measure them by. Along with accuracy, concepts that can be tested include model relevance, efficiency, robustness, fairness, and interpretability [1].

For instance, metamorphic testing is a type proposed as method of testing without using test oracles, where test cases are constructed based on defined metamorphic relations [14]. Another is of automated fairness testing, which uses symbolic evaluation techniques on explanatory techniques to evaluate a mode's fairness [15].

There is also the so-called 'trust score'; a measure of the level of agreement between a classifier and a modified nearest-neighbor classifier for a single response [16]. This is a rather different approach to our proposed technique, as it relies on processing the test data using a distance metric, its effectiveness depending on the number of dimensions.

None of these, however, measure the overarching concept of trustworthiness as a whole.

2.2 Trustworthiness testing

Trustworthiness is defined by many to be the combination of a variety of concepts, such as robustness, security, transparency, fairness, and safety [17, 18]. This is neither a clear nor complete definition, however, and does capture the essence of the concept. The definition we will be using instead is thus:

A system is trustworthy to a stakeholder in a context if: the system works properly in the context, and the stakeholder is justified in their belief of that, given they believe it [6].

'Works properly' is intentionally vague, giving room to explore the concepts in the other definition.

To note, there is a difference between trust and trustworthiness. Trust is the perception a person has towards a system, while trustworthiness is a property of that system. People can trust an untrustworthy system, and vice versa [6]. Indeed, there has been times where a more trustworthy system has had little impact on the user's trust, even though their objective task performance increased significantly [19].

Several metrics have been proposed to quantify the concept of trustworthiness. However, much of the work on trustworthiness testing, unlike the other forms of testing mentioned above, rely on human interpretation within their systems.

There is a metric is based on the increase in information transfer rate for humans doing a task – that is, the change in speed and accuracy of human interpreters when assisted by a machine explanation or not [20]. The Trustworthy Explainability Acceptance metric requires industry experts to evaluate the system in its calculations [21, 2]. There is the building of trust probabilities based on subjective logic via uncertainty given by a human observer [3]. RITUAL is a collection of metrics that require humans to manually evaluate [4].

Thus, to our knowledge, our technique will be the first to automate the evaluation of trustworthiness in a model – or, rather, trustworthiness backed by the justifications provided by explainability.

2.3 Explainability

While trustworthiness can be partially measured through a variety of metrics as mentioned before, it can be argued that explanations is one of the most crucial aspects that result in it, as it provides clear justifications for users to base their belief on [6]

The definition of explainability, as given in the most recent explainable AI survey, is that "Explainability provides insights to a targeted audience to fulfill a need" [5]. That is, the output from explainability techniques is given to such people as domain experts, end-users, and modeling experts to handle such issues as justifying decisions, discovering new knowledge, improving the black-box AI model, and ensuring fair decisions. However, there has been no exploration in the possible removal of the target audience; that is, what if the insights are given instead to a machine to evaluate?

Explainability techniques are vast and varied, including such things as saliency maps, feature attribution methods, counterfactual and contrastive explanations, white-box models, and global surrogates [22]. We will focus on techniques that provide textual output; specifically, feature attribution methods, which are techniques that identify the most important features in an input for a given prediction. They will be system-agnostic post-hoc explainability techniques (though, given the nature of these methods, they only apply to classifiers).

There are many techniques available to use [5]. LIME, one of the most popular methods, works by creating a simplified model that approximates the behavior of a more complex model for a particular input [23]. SHAP, another popular method, uses Shapely values to assign a value to each feature indicating its contribution to the prediction [24].

There are several papers describing explainability with testing. There have been testing on the effectiveness of individual explanatory techniques [25] and testing of the evaluation of explanations [26, 27]. The explanations themselves can be unreliable, so there has been research to mitigate this as well [28, 29].

More relevant to our goal, there have been papers on evaluating a model based on explanations [20], including the original LIME paper itself [23]. However, as stated in the

previous section, human involvement is almost always a requirement and there seems to have never been any research into automating this process.

To note, there is a technique briefly mentioned previously that does provide an automated method for the evaluation of fairness, via the use of the local linear model created by LIME [15]. However, fairness by itself does not constitute a full evaluation of trustworthiness; our technique focuses on evaluation based on the justification behind the explanations besides.

2.3.1 Explanations and trustworthiness

It is a common sentiment that model explanations can promote trustworthiness [30, 6]. Explanations have the properties of faithfulness – how well the explanation represents the workings of the model – and plausibility – how convincing the explanation is to a human [31]. Most papers relating to this aim to evaluate the faithfulness of explanations. This, however, is outside our scope; we assume that explanation methods work as intended.

Instead, we focus on the evaluation of the trustworthiness of a model through the proxy of the plausibility of its explanations. The current work towards the measurement of plausibility seems to mainly within the natural language processing field. In the literature, there seems to be no fully automatic method to measure plausibility. They involve procedures such as comparing against human-created ground truths [32, 33, 34], or asking people directly [35]. There are also some automated metrics, but again they only compare against human-made gold standards [36].

From a psychological perspective, there are also the concepts of comprehensiveness – whether the explanation includes all possible variables that could explain the answer – and sufficiency – where the explanation can contain only the minimum needed to explain it [37]. It

seems that sufficiency has more in line with how humans evaluate their trust for explanations [38, 35, 36].

2.4 Word embeddings

Word embeddings are vector representations of words, showing their semantic relationship, as found via their use in similar contexts. By converting this human concept into a mathematical representation, it allows one to use powerful mathematical operations to more efficiently evaluate the data. For our use, it allows for the quantification of the closeness of the conceptual relationship between words.

One aspect to note: we are transferring our dubious trust in the black-box model onto the word embedding model; but how do we know the latter can be trusted? Word embedding models are themselves models, and thus can fall prey to the same problems we are trying to use them to solve. However, there has been extensive research in this field to mitigate these concerns [7, 9, 39, 10]. We will also be using an ensemble of word embedding models, which will likely lead to a more trustworthy model [11].

3 TOWER

We now present Trustworthiness Oracle through Word Embedding Relatedness (TOWER). The goal of TOWER is to automatically determine the trustworthiness of a given model-agnostic classifier through the use of explainability techniques and word embeddings.

In summary, the process is as follows: An input is fed into the classifier, after which explanation techniques are used to produce an explanation for it. By our definition of

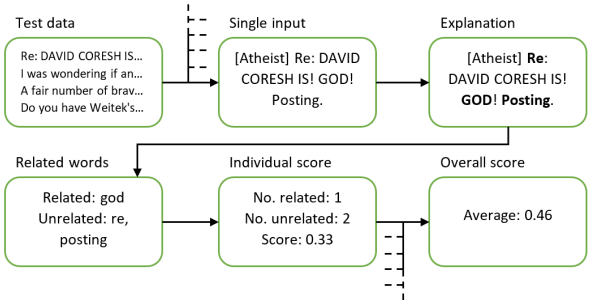

Figure 1: Overall TOWER design

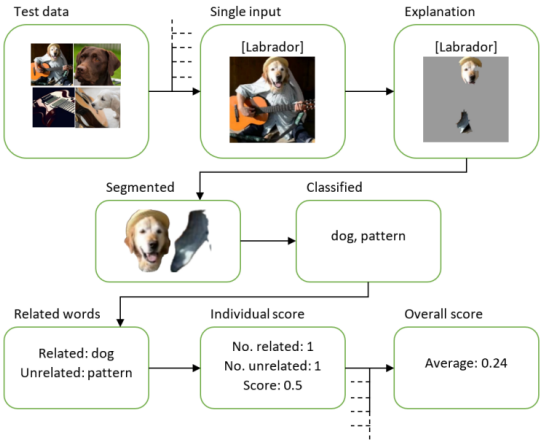

Figure 2: Image extension for TOWER

trustworthiness, if the classifier classified incorrectly, the input is thrown out. From the explanation, the most significant words are identified, then processed through a word embedding ensemble to find their relatedness to the given category. This is then used to calculate the prediction score for each input instance, which is ultimately compiled to give the model score. This can be seen illustrated in figure 1.

This can also be extended to work on images as well as text. Here, an additional step is inserted after the explanations to label the items which appear in the explanation, which is then fed into the word embedding ensemble as above. See figure 2.

More details of most of these can be found in my project partner's report. Here, I will only delve into the workings of the word embeddings segment.

3.1 Word embeddings

In this step, we take as input two variables: the word describing the category of the instance, and the explanation – the list of words that have been deemed significant in the classification of the instance. The output will be a list of 3-tuples of how related each word is to the

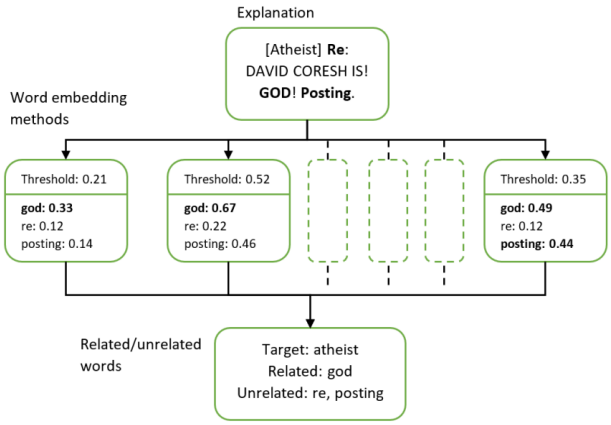

Figure 3: Design for ensemble

category, in the form (unrelated, related, indefinite).

Before input, some preprocessing is done: namely, lowercasing the text as well as lemmatizing the words.

To alleviate concerns simply moving the burden of trust from the classifier to a word embedding, we use an ensemble of various methods that perform this operation together. Via popularity on TFHub and current state-of-the-art review [40], we have chosen five: FastText, GloVe, Neural-Net Language Model (NNLM), Swivel, and Universal Sentence Encoder (USE). For simplicity, we will refer to these as 'word embeddings'.

Each word embedding takes two text inputs and produces a numerical score based on how related it regards the them as. However, different word embedding outputs cannot be directly compared to each other. Therefore, to form an ensemble, we create a threshold for each word embedding, where the texts are regarded as related when the output is above the threshold, and similarly for below [41]. This can then be combined into one score, either via an average or plurality method. This process is illustrated in 3.

To calculate this threshold, we can turn this into a binary classification problem. We can

first find related pairs of words via synonyms. We take the top 1000 most common English words from WordNet [42] then use Merriam-Webster [43] to find various synonyms of each word, for a total of approximately 32,000 pairs of similar words. Then, we generate another 32,000 random pairs from the WordNet words to create a list of non-similar pairs. Running the word embeddings on these lists, we do a binary search to find the threshold that balances the precision and recall of each.

To keep in mind, there is a difference between relatedness – how associated two words are with each other (e.g. cat, whiskers) – and similarity – a subset of relatedness on how alike the are (e.g. cat, feline). We want to find the threshold for relatedness. As synonyms are a metric of similarity and similarity is a subset of relatedness, synonyms are also a strong measure of relatedness.

We can modify this ratio by a $precisionmodifier$. A value above 1 results in a lower threshold, and the opposite for higher.

A possible concern of this binary discretisation is that this is not a good representation for values near the threshold, which could easily be on the other side. This is alleviated by having an option, $exclusionrange$, that creates a certain range around the threshold where the values are labelled as 'indefinite'.

Another concern is that the worse-performing word embedding methods could drag down the overall performance of the ensemble. To help combat this, there is an option to add a 'weighting' to each. This represents how 'good' the method is at classifying related and unrelated pairs, via an area-under-the-curve (AUC) score. This can then be used to modify how impactful each of the methods are on the final score produced by the ensemble.

4 Evaluation

In this section, we evaluate TOWER through three methods. In the first, we look at its performance on models trained on various levels of noise. In the second, we compare its results against human-produced ground truths. Finally, we compare against GPT-3.5, to see if this entire endeavour can be supplanted by a few prompts.

I will be focusing on the first of these methods in this report. The latter two will are covered in more detail in my project partner's report.

4.1 Noise-based evaluation

4.1.1 Method

The goal of this first experiment is to evaluate whether TOWER can detect a difference in trustworthiness between machine learning models, as well as find the best-performing configuration for the tool.

To do this, we assume that the higher the level of noise or bias in the dataset a model is trained on, the lower the trustworthiness of the model. Using this assumption, we train several models on datasets of different levels of artificially-induced noise, then evaluate our tool on them. This is common practice in the literature [44, 45, 46, 47, 23, 48, 49].

This evaluation can be broken into three steps: the training of the models, the use of TOWER at various configurations, and the choosing of a good analysis method.

First, we have several options for the data generation and model training:

- *Model type* — Through scikit-learn, we train the classifier models multinomial naive Bayes (MNB), decision tree, random forest, and stochastic gradient descent (SGD). We

do this through five partitions for cross-validation.

- *Dataset* — Sorting by most downloads on HuggingFace, we chose the datasets 20 Newsgroups, AG News, DBpedia 14, Emotion, and IMDB. The 20 Newsgroups dataset was also featured in the original LIME paper [23].

- *Noise type* — Let p be the noise level percentage. We chose three types of noise to consider: randomly removing 30%-70% of words from p of the texts [47]; changing the label of p of the data to a random other one [45, 47, 49]; and bias, by, for each category, choosing a line of text from a random Wikipedia article and adding it the end of p of instances of that category. We also have a fourth noise type for the 20 Newsgroup dataset – a so-called 'natural' noise – where noise is introduced by adding mostly-irrelevant headers and footers to p of the data that were originally removed. This was used in the experiment done in the original LIME paper [23].

We do this for five levels of noise: 0%, 25%, 50%, 75%, and 100%.

For testing, we take both noisy and clean versions of each instance in the test data and combine them into a larger set as our resultant test data. This is so that all models of each noise type will be tested on the same data, regardless of their noise levels.

We then have another set of configurations for the tool itself: *word_embedding_precision_modifier*, *word_embedding_exclusion_range*, *do_word_embedding_weighting*, *calculate_word_scores_method*, *explanation_threshold*, *explanation_top_n*, and *calculate_instance_scores_method*. More details of these can be found in section 3.

By running TOWER using every combination of both the generation and configuration options, we produce a resultant model score for each. This, as a reminder, is a 3-tuple in the

form of trustworthiness, untrustworthiness, and indefinite percentages; that is, (t, u, i). We now want to plot this against noise level and find the configuration combination that results in the steepest slope, and thus the strongest correlation between increasing noise level and decreasing trustworthiness, according to the tool.

Here, several questions arise. Firstly, obtaining a slope requires a single value for each data point; yet, what we have is three values for each. Next, the fact that decreasing noise levels may also decrease accuracy, which may affect results. Finally, it is likely that the noise types will not be equally good at representing untrustworthiness, especially given that they are quite different in detail.

Thus, we have yet another set of options for the analysis of these results:

- *Slope calculation type* — We can calculate the slope either by taking only the decrease in t; taking only the increase in u; or taking the decrease in t / (t + u).
- *Slope adjusted* — To reduce the impact of accuracy, the calculation can be adjusted such that it only takes into account all instances that are correctly labelled across the different noise levels.
- *Noise types* — What combination of different noise types to consider.

Thus, the overall process is as follows:

Let G, C and A represent the all possible combinations of the generation, configuration and analysis options, respectively. Take some combination of options a' in A and g' in G. For each c in C, we run TOWER on the model generated by g'. This produces a set of trustworthiness scores. We do this for each noise level. Then, for each c, we can produce a

slope of trustworthiness score against noise level to roughly approximate how well-correlated that configuration is.

Next, as these slopes cannot be directly compared between instances of G due to their inherent differences, we analyse using ranking. Within g', we sort each c by their associated slope, creating a ranking of how well the configurations perform.

We do all of this for each g in G (taking into account only the noise types chosen in a'), resulting in a set of rankings. Then, for each c, we sum their rankings and sort by that to produce an overall ranking of which instances of c has performed the best.

We then do this for each a in A, which results in a set of overall rankings. To find the best a to use, we do not do another naive rank sum, as we do not want to put equal weighting on all analysis options. Instead, we assume that the ideal a would be one that has the least change between the slope calculation methods. Thus, choosing the rankings obtained by varying this variable and fixing the others, we take the sum of the differences for how much in rank each c changes between the rankings, producing an overall score for how 'similar' the rankings are.

Finally, after choosing the set of a that vary the least between slope calculation methods, we can do one last rank sum for each c to find which configuration of TOWER functions the best.

4.1.2 Results

Through this experiment, we find that TOWER does detect some form of trustworthiness, as the slopes for most noise types generally trend down.

We find that the analysis combination that is the most similar between calculation types

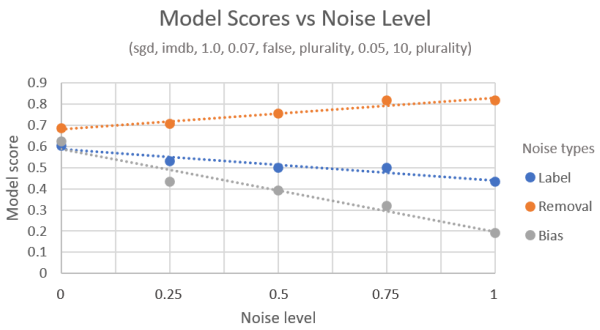

Figure 4: Example slopes for a certain configuration

is using adjusted slope, as well as using only bias as 'noise'. Thus, under the assumption that TOWER detects trustworthiness, we can say that adding bias is the method that is most representative of creating untrustworthiness.

To note, we find that label noise does not evaluate correctly with adjusted slope, as the result of having 100% label noise is 0% accuracy. Thus, all analysis options involving both adjusted slope and label noise are not considered.

Using this analysis combination, we find that the best configuration combination is: *word_embedding_precision_modifier* – 1.0, *word_embedding_exclusion_range* – 0.07, *do_word_embedding_weighting* – false, *calculate_word_scores_method* – plurality, *explanation_threshold* – 0.05, *explanation_top_n* – 10, *calculate_instance_scores_method* – plurality.

We can verify that, for the three slope calculation types, this configuration is ranked 18, 8 and 1, respectively, out of a maximum of 96.

Interestingly, we also see that, for the 20 Newsgroup dataset, removal noise and natural noise have a similar slope. However, they are both very shallow, neither seemingly having much impact on trustworthiness, despite what is shown in the LIME paper [23].

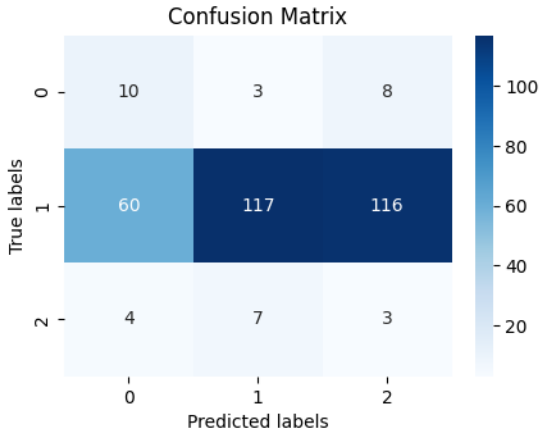

Figure 5: Confusion matrix of TOWER's first configuration against ground truth

4.2 Human-based evaluation

The goal of this second experiment is to compare TOWER against human-created ground truths, to see if it aligns with the human sense of trustworthiness.

To do this, we take eight pretrained models that have been fine-tuned on eight different databases, chosen for their popularity on HuggingFace. Then, we run these models through LIME to produce various instances of explanation data. After that, we then have humans label whether they think each instance is trustworthy, producing a ground truth of what is and is not considered a trustworthy explanation. TOWER, using the configuration from the first experiment, is then run on this data to evaluate the tool.

For results, we find that the configuration found in the previous experiment does not match well with the gold standard created in this experiment. We can see in figure 5 that, while running TOWER gives a roughly equal distribution of trustworthy-untrustworthy-indefinite, the human labels give an approximate 6%/89%/4% split.

More details of this experiment can be found in my project partner's report.

4.3 GPT

The goal of this third experiment is to see how TOWER compares with GPT-3.5. We did this by using a few-shot method to prompt GPT-3.5-turbo to give trustworthiness scores for each ground truth instance produced from the last evaluation. The result is that GPT seems to perform worse than TOWER, thankfully.

Again, more details can be found in my project partner's report.

4.4 Discussion

From these results, we find that TOWER is, in fact, very ineffective at what it does.

Looking at the instances that TOWER labels incorrectly, we can draw some conclusions.

Out of the 328 ground-truth instances, TOWER labels 198 incorrectly. Of those, 135 are when either the ground-truth or TOWER labels are 'indefinite'. We will put these aside for now, as we consider the correct labelling of 'trustworthy' and 'untrustworthy' more important. Of the 63 remaining, 60 are false negatives. Of the 63, there are 32 from the Amazon Polarity, Rotten Tomatoes, and IMDB datasets – the ones with the categories 'positive' and 'negative'. The rest are variously from the others, the most common categories being 'joy', 'sports', 'science', 'sadness', and 'health'.

From this, we can intuit that the effectiveness of TOWER is somewhat proportional to how specific the category word is. The more general the category, the harder it is to relate words to it. To alleviate this, a possible solution is to have the category instead be a sentence describing the concept in more detail, such as a definition.

TOWER also does not take into account various human thought processes that may occur

when deciding if an explanation is trustworthy or not. For example, when a cluster of words are in the explanation (e.g. "not good"), or when a single word has an unusually high impact on the input.

The results also imply that using noise in the first evaluation is actually not a good approximation of trustworthiness from a human perspective. More resarch would be needed in this area as well.

5 Conclusion and future work

This paper presents TOWER, an automated generator of trustworthiness oracles for machine learning classifiers. Our experiments indicate that, while the concept intuitively seems sound, there is much to be done in terms of its effectiveness.

There are many avenues of future work that can be done. Firstly, to improve TOWER's efficacy, research into sentence-based category descriptors can be done, as well as context-based embeddings and taking explanations as a whole. This paper assumes that the explanations are completely faithful; this is likely not the case. Other explanation methods such as SHAP can be explored to improve this. The image extension of TOWER, while implemented, has not been evaluated; this can also be done. Finally, a further exploration of what it means for a person to find an explanation trustworthy would be very useful in the creation of similar systems to TOWER in the future.

References

[1] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning testing: Survey, landscapes and horizons," in *IEEE Transactions on Software Engineering*, pp. 1–1, 2020.

[2] D. Kaur, S. Uslu, A. Durresi, S. Badve, and M. Dundar, "Trustworthy explainability acceptance: A new metric to measure the trustworthiness of interpretable ai medical diagnostic systems," in *Complex, Intelligent and Software Intensive Systems* (L. Barolli, K. Yim, and T. Enokido, eds.), (Cham), pp. 35–46, Springer International Publishing, 2021.

[3] M. Cheng, S. Nazarian, and P. Bogdan, "There is hope after all: Quantifying opinion and trustworthiness in neural networks," *Frontiers in Artificial Intelligence*, vol. 3, 2020.

[4] A. H. CeldrÃ¡n, J. Bauer, M. Demirci, J. Leupp, M. F. Franco, P. M. SÃ¡nchez SÃ¡nchez, G. Bovet, G. M. PÃ©rez, and B. Stiller, "Ritual: a platform quantifying the trustworthiness of supervised machine learning," in *2022 18th International Conference on Network and Service Management (CNSM)*, pp. 364–366, 2022.

[5] W. Saeed and C. Omlin, "Explainable ai (xai): A systematic meta-survey of current challenges and future opportunities," *Knowledge-Based Systems*, vol. 263, p. 110273, 2023.

[6] L. KÃ€stner, M. Langer, V. Lazar, A. SchomÃ€cker, T. Speith, and S. Sterz, "On the relation of trust and explainability: Why to engineer for trustworthiness," in *2021 IEEE 29th International Requirements Engineering Conference Workshops (REW)*, pp. 169–175, 2021.

[7] T. Bolukbasi, K.-W. Chang, J. Zou, V. Saligrama, and A. Kalai, "Man is to computer programmer as woman is to homemaker? debiasing word embeddings," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, (Red Hook, NY, USA), p. 4356â4364, Curran Associates Inc., 2016.

[8] F. Torregrossa, R. Allesiardo, V. Claveau, N. Kooli, and G. Gravier, "A survey on training and evaluation of word embeddings," vol. 11, p. 85â103, 2021.

[9] B. Wang, A. Wang, F. Chen, Y. C. Wang, and C.-C. J. Kuo, "Evaluating word embedding models: methods and experimental results," *APSIPA Transactions on Signal and Information Processing*, vol. 8, 2019.

[10] R. Schuster, T. Schuster, Y. Meri, and V. Shmatikov, "Humpty dumpty: Controlling word meanings via corpus poisoning," in *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 1295–1313, 2020.

[11] S. Kumar, P. Kaur, and A. Gosain, "A comprehensive survey on ensemble methods," in *IEEE 7th International conference for Convergence in Technology*, 2022.

[12] E. T. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo, "The oracle problem in software testing: A survey," *IEEE Trans. Softw. Eng.*, vol. 41, p. 507â525, may 2015.

[13] G. Canbek, T. T. Temizel, and Åeref SaÄiroÄlu, "Ptopi: A comprehensive review, analysis, and knowledge representation of binary classification performance measures/metrics," *Sn Computer Science*, vol. 4, 2022.

[14] X. Xie, J. W. K. Ho, C. Murphy, G. Kaiser, B. Xu, and T. Y. Chen, "Testing and validating machine learning classifiers by metamorphic testing," *J. Syst. Softw.*, vol. 84, p. 544â558, apr 2011.

[15] A. Aggarwal, P. Lohia, S. Nagar, K. Dey, and D. Saha, "Black box fairness testing of machine learning models," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2019, (New York, NY, USA), p. 625â635, Association for Computing Machinery, 2019.

[16] H. Jiang, B. Kim, M. Y. Guan, and M. Gupta, "To trust or not to trust a classifier," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, (Red Hook, NY, USA), p. 5546â5557, Curran Associates Inc., 2018.

[17] J. M. Wing, "Trustworthy ai," vol. 64, (New York, NY, USA), p. 64â71, Association for Computing Machinery, sep 2021.

[18] B. Li, P. Qi, B. Liu, S. Di, J. Liu, J. Pei, J. Yi, and B. Zhou, "Trustworthy ai: From principles to practices," vol. 55, (New York, NY, USA), Association for Computing Machinery, jan 2023.

[19] A. Schmitt, T. WambsganÃ, and A. Janson, "Designing for conversational system trustworthiness: The impact of model transparency on trust and task performance," in *Thirtieth European Conference on Information Systems (ECIS)*, 06 2022.

[20] P. Schmidt and F. Biessmann, "Quantifying interpretability and trust in machine learning systems," vol. abs/1901.08558, 2019.

[21] S. Uslu, D. Kaur, S. J. Rivera, A. Durresi, M. Durresi, and M. Babbar-Sebens, "Trustworthy acceptance: A new metric for trustworthy artificial intelligence used in decision making in food–energy–water sectors," in *Advanced Information Networking and Applications* (L. Barolli, I. Woungang, and T. Enokido, eds.), (Cham), pp. 208–219, Springer International Publishing, 2021.

[22] F. Cabitza, A. Campagner, G. Malgieri, C. Natali, D. Schneeberger, K. Stoeger, and A. Holzinger, "Quod erat demonstrandum? - towards a typology of the concept of explanation for the design of explainable ai," *Expert Systems with Applications*, vol. 213, p. 118888, 2023.

[23] 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*, KDD '16, (New York, NY, USA), p. 1135â1144, Association for Computing Machinery, 2016.

[24] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 4768â4777, Curran Associates Inc., 2017.

[25] J. Zhou, A. H. Gandomi, F. Chen, and A. Holzinger, "Evaluating the quality of machine learning explanations: A survey on methods and metrics," *Electronics*, 2021.

[26] S. Mohseni, J. E. Block, and E. Ragan, "Quantitative evaluation of machine learning explanations: A human-grounded benchmark," in *26th International Conference on Intelligent User Interfaces*, IUI '21, (New York, NY, USA), p. 22â31, Association for Computing Machinery, 2021.

[27] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," in *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 80–89, 2018.

[28] P.-J. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, and B. Kim, *The (Un)Reliability of Saliency Methods*, p. 267â280. Berlin, Heidelberg: Springer-Verlag, 2022.

[29] A.-K. Dombrowski, M. Alber, C. J. Anders, M. Ackermann, K.-R. Müller, and P. Kessel, *Explanations Can Be Manipulated and Geometry is to Blame*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[30] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.," vol. 16, (New York, NY, USA), p. 31â57, Association for Computing Machinery, jun 2018.

[31] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," in *Conference on Empirical Methods in Natural Language Processing*, 2019.

[32] J. DeYoung, S. Jain, N. Rajani, E. P. Lehman, C. Xiong, R. Socher, and B. C. Wallace, "Eraser: A benchmark to evaluate rationalized nlp models," in *Annual Meeting of the Association for Computational Linguistics*, 2019.

[33] S. Chang, Y. Zhang, M. Yu, and T. Jaakkola, "Invariant rationalization," in *Proceedings of the 37th International Conference on Machine Learning* (H. D. III and A. Singh, eds.), vol. 119 of *Proceedings of Machine Learning Research*, pp. 1448–1458, PMLR, 13–18 Jul 2020.

[34] J. Bastings, W. Aziz, and I. Titov, "Interpretable neural predictions with differentiable binary variables," vol. abs/1905.08160, 2019.

[35] S. Jain, S. Wiegreffe, Y. Pinter, and B. C. Wallace, "Learning to faithfully rationalize by construction," in *Annual Meeting of the Association for Computational Linguistics*, 2020.

[36] M. Clinciu, A. Eshghi, and H. Hastie, "A study of automatic metrics for the evaluation of natural language explanations," vol. abs/2103.08545, 2021.

[37] S. Carton, A. Rathore, and C. Tan, "Evaluating and characterizing human rationales," in *Conference on Empirical Methods in Natural Language Processing*, 2020.

[38] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial Intelligence*, vol. 267, pp. 1–38, 2019.

[39] L. K. Åenel, Ä. Utlu, F. ÅahinuÃ§, H. M. Ozaktas, and A. KoÃ§, "Imparting interpretability to word embeddings while preserving semantic structure," *Natural Language Engineering*, vol. 27, no. 6, p. 721â746, 2021.

[40] M. Mars, "From word embeddings to pre-trained language models: A state-of-the-art walkthrough," *Applied Sciences*, vol. 12, no. 17, 2022.

[41] L. Mariani, M. PezzÃš, V. Terragni, and D. Zuddas, "An evolutionary approach to adapt tests across mobile apps," in *2021 IEEE/ACM International Conference on Automation of Software Test (AST)*, pp. 70–79, 05 2021.

[42] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, vol. 38, p. 39â41, nov 1995.

[43] Merriam-Webster, "Api," 2023. Merriam-Webster's Online Dictionary, API version.

[44] X. Zhu, X. Wu, and Q. Chen, "Eliminating class noise in large datasets," in *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, p. 920â927, AAAI Press, 2003.

[45] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study of their impacts," *Artif. Intell. Rev.*, vol. 22, p. 177â210, nov 2004.

[46] S. Agarwal, S. Godbole, D. Punjani, and S. Roy, "How much noise is too much: A study in automatic text classification," in *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pp. 3–12, 2007.

[47] E. Apostolova and R. A. Kreek, "Training and prediction data discrepancies: Challenges of text classification with noisy, historical data," *ArXiv*, vol. abs/1809.04019, 2018.

[48] D. Nguyen, "Comparing automatic and human evaluation of local explanations for text classification," in *North American Chapter of the Association for Computational Linguistics*, 2018.

[49] I. Jindal, D. Pressel, B. Lester, and M. Nokleby, "An effective label noise model for DNN text classification," 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)*, (Minneapolis, Minnesota), pp. 3246–3256, Association for Computational Linguistics, June 2019.

Abstract

In machine learning, there is an issue called the test oracle problem. Due to the nature of how models are trained, and what their purpose is, there is no certain way to define a test oracle that covers all possible situations, hampering the ability to test models effectively. One aspect of machine learning that people have tried to test is trustworthiness. Trustworthiness suffers significantly from the test oracle problem, as it seems the only possible test oracle at times are end-users, which aren't always available for testing. To fix this, we introduce TOWER, an automatic method that attempts to define a trustworthiness oracle for testing purposes. TOWER uses explanation tools, such as LIME, and word embedding models to test word classifier models for trustworthiness.

1 Introduction

Machine learning has come a long way in recent years, and the amount of research invested into this area has been growing. Machine learning testing in particular has received a lot of attention recently, with the number of research papers increasing rapidly between 2017 to 2019 [1]. This research has been dominated by testing metrics such as accuracy, but other metrics, such as trustworthiness, have an important role in testing. Trustworthiness usually refers to a combination of factors that affect how people trust a model, including robustness, security, transparency, fairness, and safety [2].

Recently, models such as ChatGPT have begun to have a massive impact in varies industries, and its potential continues to grow as new large language models develop. Around this growth, ChatGPT has generated a lot of attention on whether to trust these models,

and by how much, as their limitations get revealed, and potential for misinformation. This highlights the importance of trust for these models, and artificial intelligence in general.

In the context of this project, trustworthiness is related to the measure of how much a machine learning model's method of obtaining an answer is trustworthy to humans, which is heavily related to explainability. Explainability is the measure of a model's ability to explain to users their process of obtain answers. Highly explainable models, such as white box models, will more likely be trusted by users compared to a black box models that performs similarly. Trustworthiness, in this case, is measured by users. Tools can be used to obtain explanations of black box models [3] [4], which users can then use to determine trustworthiness. In these cases, users are the test oracle, they are the ones who determine whether a model is trustworthy or not.

These methods of testing trustworthiness are not automatic. They rely on a user determining the final assessment of a model, which can be a lot more time consuming and expensive compared to an automatic system. There are currently no automatic testing methods for trustworthiness, and the barrier for this seems to be the inability to determine trustworthiness without humans. A method that can automatically define a trustworthiness test oracle could decrease the time and effort required for creating trustworthy machine learning models. It would no longer be required for humans to determine the trustworthiness of a model, which can be time consuming and expensive compared to an automated method that achieves the same end. This method could also be used to train trustworthy models through reinforcement learning based of their trustworthiness. If all of these were to be achieved, the ability to create trustworthy machine learning models would increase, which may lead to benefits in trust between these models and their users.

To tackle this problem, we introduce TOWER, a novel method that tries to automate the testing process for trustworthiness. TOWER uses explainability, such as LIME [3] snd SHAP [4], that can extract the important features a model used to get its answer, and then use a word embedding tool to determine how related those features are to the predicted answer. Along with TOWER, we also contribute two datasets, one of LIME explanations labelled as trustworthy or not by humans, and one with instances for text classification taken from other datasets with noise added to them.

2 Literature Review

There is five main groups of relevant literature that was highlighted for this project, which are categorised as: machine learning testing, trustworthiness testing of machine learning models, machine learning explainability, testing using word embedding, and Plausibility, Faithfulness and Sufficiency.

2.1 Machine Learning Testing

Machine learning testing is a lot more difficult compared to software testing, with new challenges arising due to the fundamentally different nature and construction of machine learning systems [1]. This makes it challenging to apply existing software testing methods on machine learning systems. Testing models is also made difficult by the fact that in use, models may be answering questions that do not have a previous answer [5], which makes it hard to understand how accurate these answers will be. In addition, testing is often done on the whole model at once, making identifying where problems occur a lot harder.

To test a machine learning model, a test oracle is required to determine whether the performance of a model is satisfactory or not. The problem with machine learning testing is defining a test oracle. The end users of a model may be the test oracle, or a test oracle could be a required threshold a model needs to obtain in a certain metric. For both automated and manual test oracles, there are several tools and methods that can be utilised to improve their ability to evaluate models.

Some tools help select cases where the answer is wrong, which can help the tester understand which cases fail and potentially draw an understanding of why they fail [6]. Explainability tools help testers, especially those who may not have much technical knowledge, understand models better, which may lead to insights in how to improve them [3] [4]. Some models can generate inputs from an original input to test the model, and potentially look for corner cases through specifically generated inputs to test the model's ability at the extremes [7].

2.2 Trustworthiness testing

When looking for trustworthiness testing, the topic that this project attempts to tackle, there weren't any papers that try to automate this testing. A potential reason for this may be due to how hard it can be to quantify a metric for trustworthiness without using a human, and the difficulty in getting human-like results from an automated process. There are two major methods found that attempt to test trustworthiness, a tool called RITUAL [8], and an algorithm that generates a trust score on an answer [9]. There are also tools that require user input to calculate trustworthy metric [10] [11], but because these tools require user input,

they are not relevant tools for this project.

RITUAL generates a trustworthiness score on a supervised machine and deep learning models. It uses four main metrics to generate this score: fairness, explainability, robustness, and accountability. These metrics aggregate to create the score. With this method, trustworthiness is divided into four other categories, whereas for this project, trustworthiness will to its own category, separate from these four concepts.

The trust score algorithm generates a trust score for a prediction made by a classifier model, the same type of model this project is focused on. This is done by comparing the distance the test sample is from being predicted as the class that was predicted and a different class. If an input is close to two different classes, then it is less certain which class the input belongs to, and so the prediction shouldn't be trusted as highly. This differs from our objective as it calculates a trust score for predictions made, whereas our goal is to evaluate the trustworthiness of the whole model.

From these, the primary gap that has been identified in trustworthiness testing is the lack of methods that can measure if the process a model takes to obtain an answer is trustworthy to a human. Our idea is to use a tool that can get an explanation of this process, and then use word embedding and other potentially useful tools to determine if this process is trustworthy. As far as we have researched, this has not been attempted in machine learning.

2.3 Machine learning explainability

Explainability tools are important for this project. They are central to how we want to test trustworthiness, as we will be directly evaluating the process that the explainability tool

acquires to determine trustworthiness.

Local explanation vectors [12] can be used to signify how an individual prediction could be changed by changing a certain aspect of the input. This method does give insight into the process of prediction, as when an answer is wrong, you can see what features the input would need for the answer to be predicted as correct, but it may not be suitable for using word embedding on, and so isn't a priority for this project.

Explainability tools that can comply the process into a list of contributions, and how those contributions affect the outcome of a model, are ideal for this project [3] [4] [13]. This would allow us to use word embedding on these contributors so reveal if they are related to each other, or the concept of the class being predicted, to determine if the contributors make sense as contributors. The two tools that we think could be used to great effect for our method are LIME [3] and SHAP [4], both of which are very influential methods.

Both LIME and SHAP are able to effectively explain the decision classifier models to users, and both would be an acceptable tool for this project due to its ability to provide the contributors to answers. There is also the potential for the use of extensions [13] that may improve the performance of these models, and even combining and using both models [14], to achieve more reliable and accurate explanations not dependent on one tool or the other.

2.4 Word Embedding

Word embedding is the primary candidate to use to evaluate trustworthiness instead of a human user. The idea is that we can use word embedding to determine how related contributes are to each other, and to the class that was predicted, by using word vectors.

The closer the word vectors are to relevant concepts, the more likely the contributors are related, and thus the more trustworthy the model's process in obtaining a prediction is.

Word embedding can be prone to biases due to the data used to train them [15] [16], which has the potential to heavily impact our methods ability to measure certain models that relate to any bias it may have. To counter this, our method may employ ensemble methods [17], that is using multiple word embedding to make it more reliable as we don't rely on the single word embedding model.

Word embedding has been used in methods in some areas, including automated essay scoring (AES) [18]. Word embedding in these methods are used for their ability to represent word semantics and can be used to train a neural AES model [18]. Some word embedding models are specifically trained for AES and can do a better job for this purpose than generic ones, which has improved these neural AES models.

Currently there seems to be no method that utilises word embedding for machine learning testing. Word embedding isn't generally used for this purpose, and instead is used with machine learning models for some other purpose. This makes our method for evaluating the process of model predictions novel and adds the challenge of having to come up with how to implement this in our method.

2.5 Plausibility, faithfulness and sufficiency

When looking into explainability and how humans evaluate the trustworthiness of models, we found three important concepts, plausibility, faithfulness, and sufficiency. Faithfulness is the ability of a model to accurately represents the reasoning process behind a prediction [19].

This concept is less important for us as we assume LIME is already faithful. Plausibility is described as how convincing an interpretation is to a human [19]. This relates to us as we effectively want to measure the LIME explanation's plausibility to determine trustworthiness. Sufficiency for us can be described as the measure of how well a human is able to guess the correct class given the explanation [20]. This is somewhat important for us as an explanation will need to be sufficient for it to be trustworthy, however this somewhat overlaps with plausibility.

This concepts are important when forming our method for defining a trustworthiness test oracle. Ideally, plausibility will be measured, which is assumed to be related to trustworthiness, as a plausible prediction will make sense to humans, which would mean they would trust it more than if it didn't make sense. Sufficiency is questioning how much of the explanation from LIME we will need to take in order to measure for trustworthiness, which will need to be figured out.

3 TOWER

TOWER utilises explainability tools, mainly LIME, and word embedding in a pipeline that takes in a model and a dataset of instances to use on that model, and outputs an overall score of trustworthiness for the model, with the pipeline shown in figure 1. The single inputs from the dataset are put through the explainability tool, along with the model being tested, to obtain words the model used to obtain a prediction. These words are then compared with the predicted class using word embedding to find the ratio between related and unrelated words, which will be used to calculate the prediction score for trustworthiness, and then the

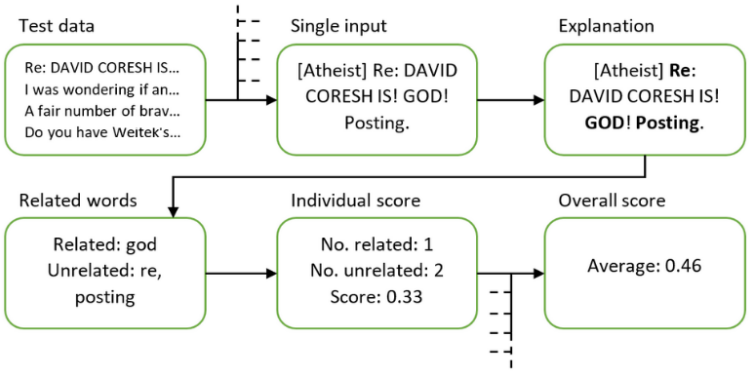

Figure 1: Pipeline of TOWER

model score for trustworthiness.

3.1 Input

TOWER can only test classification models. LIME is restricted to regressor or classifier models, and because we are basing trustworthiness on explainability, and using LIME in particular, we cannot go beyond those bounds. Classifier models predict the class of a given input, and TOWER uses that class with the words from LIME explanation to compare the two together. This means if the class being predicted is arbitrary, such as classifiers that predicting the star rating reviews were given, or if a statement is true or false, wouldn't work well with TOWER.

The names of these classes are thus an important aspect of whether TOWER can accurately determine trustworthiness. This further limits suitable classifiers TOWER can evaluate. Because word embedding is used, some class, such as classes with multiple words, hyphenated words, or words not present in major dictionaries, become unsuitable or inaccurate for TOWER. Therefore, classifiers with single word classes with concrete meaning, such as cat, dog, movie, etc., are the suitable inputs for TOWER.

3.2 LIME

LIME is a central part of TOWER. It is a popular technique used to get an explanation for a prediction made by a classifier or regressor model. The explanations include, for text classifiers, words and an associated score based on the significance that word has on the prediction, summing up to 1. There are both positive and negative scores, positive scores represent how important the word is for predicting that class, the negative scores for predicting another class. SHAP is another tool that outputs a very similar explanation, and is implemented in TOWER, however due to its speed compared to LIME and our time restraints we did not use it as our default.

If the model being tested correctly predicted the class of an input, then LIME will be used to get an explanation of that prediction. If the model incorrectly predicted the class, then this prediction is skipped and a score not given for it.

When using LIME, we had to decide how we would use the explanations it gives us, and what we would take from them. For words, LIME has a limit that decides how many words it will pick for the explanation. We decided to set this limit to 40, which seems high enough to pick out all of the important words, and have a user configuration that will take the top n number of words from that initial 40. This seems sufficient enough to make sure all important words are found. We also decided to disregard words with negative score, words that are considered important for the model picking a different class. This is because those words are dependant on what the other classes are, which is out of scope when measuring relatedness of the predicted class.

This means we would have a list of words, with positive scores, from LIME. Depending on

the explanation, there may be only a few important words, with the rest being insignificant. To combat having insignificant words, and potentially skewing the number of unrelated words, there is also a threshold on the score of the words. This threshold is a user configuration with a default value obtained through experimentation. This grants us a list of words that scored high on LIME's explanation, which is then brought to the word embedding part of TOWER.

3.3 Word embedding

TOWER uses word embedding to measure how related the words from the LIME explanation are to the class name. Effectively, all words in the word embedding models are represented as vectors. Using these vectors, you can measure how related two words are using cosine coefficient. A threshold is then used to determine if the cosine coefficient indicates whether two words are related or not. Each word is labelled as either related, unrelated, or indefinite, or averaged out, through a 3-tuple. For more details of this section, refer to my partners report.

3.4 Trustworthiness scores

TOWER calculates the trustworthiness score of each model based on the scores of every prediction being tested. Using the words from the LIME explanation and it's tuple of relatedness, every prediction score is calculated as a 3-tuple, with the indexes representing untrustworthy, trustworthy, and indefinite respectively. If a word is considered unrelated to the class, then that will decrease the trustworthiness of a prediction, and if it is related, then it will increase the trustworthiness. The final score is an averaged out 3-tuple from all the

prediction scores. There are three methods to calculate the prediction score, all being an option in the configuration, which are average, plurality and sufficiency.

Average simple outputs an tuple that is the average of the relatedness tuples for each word. For example, if there are 3 words that are related (represented as a tuple of (0,1,0)), 2 that are unrelated (1,0,0), and 1 that we are indefinite of (0,0,1), then the prediction score will be a 3-tuple of (0.5,0.33,0.17).

Plurality calculates the average of all the relatedness tuples, and then labels the prediction as trustworthy (0,1,0), untrustworthy (1,0,0), or indefinite (0,0,1), based on whatever index of the 3-tuple is highest. If there are two tuple indexes that are the highest, then the output will be indefinite.

Sufficiency calculates the average of all the relatedness tuples, and then labels predictions as trustworthy if the middle index is above 0, meaning that one of the words are related to the class. If there are no related words, then the next biggest index is used to label the prediction. This option is included due to the fact that people may only need one relevant reason [21].

3.5 Images

Because LIME also works on images, it is also possible to apply TOWER on image classifiers if we can use LIME's explanation to get words. This requires an intermediate step between getting the LIME explanation and passing the words to the word embedding. Do to this, we use a tool called Recognize Anything [22]. LIME's explanation for images involves outputting a saliency map on the image for which pixels are relevant for the model's prediction, and using

a configuration option to take out parts of the image that are low in the saliency map, leaving an image with only the significant parts for the prediction left in. For the configuration, we use the same value as is used in the paper that introduces LIME [3]. We then use Recognize Anything, which takes the image and outputs tags based on what is in the image, for example, if there is a dog in the image the tool should output dog as one of the tags. These words are then brought to the word embedding to measure the relatedness between them and the class.

Compared to testing word classifiers this method is less efficient in testing for trustworthiness as taking words from images can be inaccurate and may not be a true representation of the LIME explanation. Due to constraints in time, we will not be evaluating TOWER using images, however it is implemented in the tool and can be used, albeit in a less flexible way.

4 Evaluation

After creating TOWER, we wanted to evaluate it and run experiments to figure out how well it performs and what configuration it should use. We came up with three evaluations. For the first evaluation, we would get models that were trustworthy and models that were untrustworthy, and compare the score TOWER gives for those models across different combinations of configuration options to determine the best combination for our model. Details on this evaluation can be found in my partners report. For the second evaluation, we would run TOWER, using the configuration options obtained in the first evaluation, to get instances that were labelled and compare that with those same instances labelled by humans. For the third evaluation, we would get ChatGPT to label some instances and compare that with both TOWER and human labels to see if ChatGPT can outperform TOWER.

4.1 Second evaluation

The second evaluation has the objective of comparing how TOWER labels individual predictions made by models for trustworthiness with human labelling. TOWER takes a LIME explanation and uses it to determine trustworthiness, and so the idea is that we can use a human to get the LIME explanation and also decide if it is trustworthy, and compare the two together. For this, we assumed the human labelling was a ground truth for whether a prediction was trustworthy or not, and compared this to TOWER when using the configuration options from the first evaluation and finding the configuration options that matches the human labelling the most. Because we couldn't find any datasets that were suitable for this evaluation, we decided on creating one ourselves.

To achieve this, we needed to gather dataset instances for text classification and text classifier models that were trained on those datasets. We then needed to have TOWER label those instances as trustworthy, untrustworthy, or indefinite, and then give those same instances to humans to label.

For both the datasets and models, we went through Huggingface, and sorting by most downloads, looked through the dataset section, filtering for text classification. We choose datasets that had fine-tuned models associated with them with concrete class names that TOWER would be able to effectively test trustworthiness on. The datasets we decided on were ag news and Amazon reviews [23], DBpedia 14 [24], emotions [25], imbd [26], Stanford Sentiment Treebank [27], Yahoo Answers Topics, and news groups [28].

After we had the dataset and models, we ran TOWER, using the configuration options obtained from the first evaluation, on 1000 instances for each dataset and associated model

to get LIME explanations for those predictions, as well as the trustworthiness labels from TOWER. With this, we took predictions labelled as trustworthy, untrustworthy, and indefinite by TOWER at a ratio of 33+-7.5% to get a random distribution of those three labels, whiles making sure we still had a decent amount of each label. This was done to prevent the situation where most labels were trustworthy, which would skew the results of human labelling.

After we had the instances with LIME explanations we started human labelling. The method for this was to give a dataset instance and the possible classes and have the participant choose which class they found the instance belonged to. After they choose a class, they are shown the LIME explanation of a model's prediction, which they then label as trustworthy, untrustworthy, or indefinite. Having the participant choose the class before showing the explanation makes them think of their own explanation, which then could be used to compare with the LIME explanation. It avoids the participant mindlessly picking a label without thinking about it, which would improve them accuracy of human labelling.

Due to time constraints, we decided on labelling the data ourselves. This does bring the risk of introducing bias to the labelling process, as we know how it works and what its purpose is, however we weren't in a position to get anyone else to do the labelling. We both did as many labels as possible, making sure to do the same labels as each other. If one of us choose the wrong class, then that instance was disregarded, and if we both choose a different label then the supervisor became the tiebreaker. If he choose the class wrong or choose a different label, then the instance was also disregarded. After this process, we could then use these instances and labels as a ground truth and evaluate TOWER.

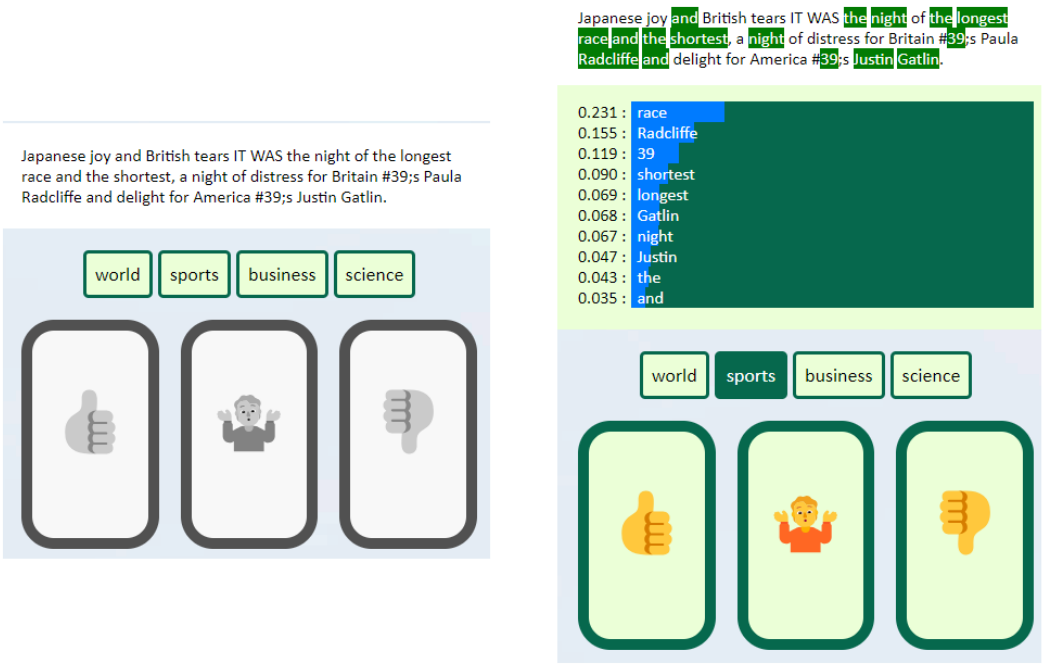

Figure 2: Second Evaluation. Left is before a class is chosen. Right is after a class is chosen, showing the LIME explanation.

4.1.1 Results

After labelling, we had 500 instances which was cut down to 328. From this, 90% of labels were trustworthy. Using the configuration from the first evaluation, we find that the precision of TOWER for labelling as trustworthy is 0.92, recall is 0.4, and f1 score is 0.56. For labelling as untrustworthy, these scores are 0.135, 0.476, and 0.21 respectively.

After getting these results, we reran TOWER using the different configuration options used in the first evaluation, as well as adding new configuration combinations that were thought to in the human labelling better. We then ranked them by the combined f1 scores for all three labels to find the best fit combination. It is important to note that this over-fits the model to the data. These new configurations score a lot better, having precision, recall, and f1 scores of 0.9, 0.97, and 0.95 for labelling as trustworthy and 0.5, 0.24, and 0.32 for

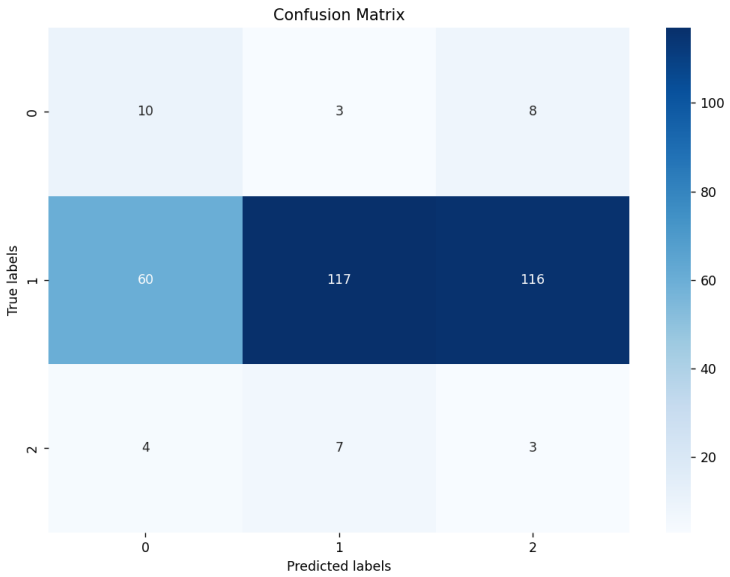

Figure 3: Confusion matrix for TOWER using first evaluation configuration. Label 0, 1 and 2 are untrustworthy, trustworthy, and indefinite respectively.

labelling as untrustworthy.

4.2 ChatGPT evaluation

To evaluate ChatGPT on the same data as evaluation two, we decided on making ChatGPT perform three tasks. The first task was to give ChatGPT an instance, the predicted class, and the confidence score of that prediction, and ask for a label. This compares ChatGPT to TOWER in its ability to determine trustworthiness of a model. The second task is the same as the first task expect we also give the LIME explanation. This compares ChatGPT to the second halve of TOWER, after we run the model through LIME and get an explanation. The third task is simulating the human labelling, giving ChatGPT text and asking it to choose a class, and then if the class is correct giving the LIME explanation and asking it to label the trustworthiness. This compares ChatGPT to the ground truth directly.

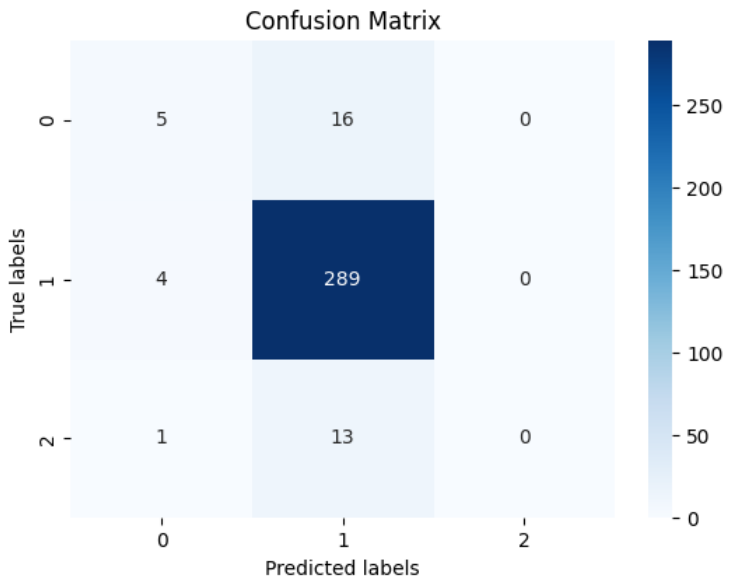

Figure 4: Confusion matrix for TOWER using new configuration that over-fits data. Label 0, 1 and 2 are untrustworthy, trustworthy, and indefinite respectively.

4.2.1 Results

For task 1, ChatGPT scored 0.91 on precision, recall, and f1 on trustworthy labels and 0.08 on precision, 0.05 on recall, and 0.6 on f1 score on untrustworthy labels. For task 2, ChatGPT scored 0.91 on precision, 0.82 on recall, and 0.86 on f1 score on trustworthy labels and 0.1 for precision, 0.29 for recall, and 0.15 for f1 score on untrustworthy labels. For task 3, ChatGPT scored 0.94 on precision, 0.83 for recall, and 0.88 for f1 score on trustworthy labels and 0.15 for precision, 0.21 for recall, and 0.15 for f1 score on untrustworthy labels.

5 Discussion

5.0.1 Second evaluation

The second evaluation has very skewed data, having 90% of labels being trustworthy. This limits the effectiveness of using this data for evaluation, as when looking for a best fit

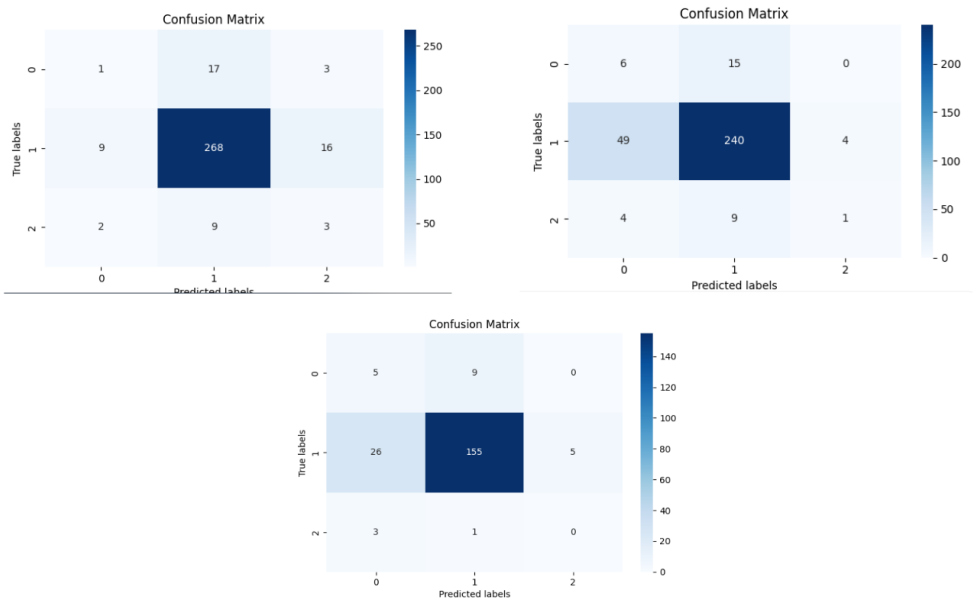

Figure 5: Confusion matrix for ChatGPT. Left for task 1, right for task 2, bottom for task 3.

configuration combination, the ones that will rank high will have a high bias towards labelling predictions as trustworthy. Despite this, due to time constraints, we still used the data for evaluation.

The first configuration options did not perform well with human labelling. The only redeeming factor is that when TOWER labels a prediction as trustworthy, then it is mostly likely trustworthy, however this does sacrifice a lot of other trustworthy predictions. Out of the 328 human labels, TOWER labels 198 incorrectly. Of those, 135 are when either the human labels or TOWER labels are indefinite. The indefinite labels are not as important as the trustworthy and untrusworthy labels, as the meaning of indefinite is different between humans and TOWER. Of the 63 remaining, 60 are false negatives. Of the 63, there are 32 from the Amazon Polarity, Rotten Tomatoes, and IMDB datasets, which have classes named positive and negative. The rest are variously from the others, the most common categories being joy, sports, science, sadness, and health.

The results show that TOWER has a bias for not labelling predictions as trustworthy. This seems to be, in part, due to the limited method of measuring trustworthiness. TOWER only measures how related words are to the class name. During human labelling, we found that we used other methods in conjunction with this to come to a conclusion. This included using process of elimination on classes if there was not a clear class that represented the text, even if the words weren't directly related to the class name, considering the LIME scores in labelling, labelling as untrustworthy if the explanation heavily relied on one word, and considering words in their context. These methods could be integrated into TOWER which may provide closer results to human labelling. TOWER could look more into the context words are placed in by combining words that are next to each other in the text and using sentence word embedding models to measure the relatedness to class names. TOWER could also look at other classes instead of just the predicted class, look at incorrect predictions and try figure out reasons why models get them wrong, and compare LIME scores for words to determine if some words are too heavily relied to be trustworthy.

The second configuration over-fitted for the human labelling is very accurate for labelling as trustworthy, but inaccurate for labelling as untrustworthy. Because the data is skewed, with 90% of labels being trustworthy, this means that the best fit configuration simply labels as many predictions as trustworthy as possible, which isn't great as a result for this evaluation. To find a better configuration, we would need to have a less skewed dataset, which was not possible due to time constraints.

5.1 ChatGPT evaluation

Using the results from the ChatGPT evaluation, comparing ChatGPT to TOWER on overfitted configurations show TOWER is a lot better at labelling untrustworthy predictions and slightly outperforms ChatGPT on labelling trustworthy predictions. The reason for ChatGPT performing better in task 1 seems to be because ChatGPT simily labelled most things as trustworthy due to the limited amount of information provided, which worked well with the skewed data. When given more information, ChatGPT ends up labelling more trustworthy labels as untrustworthy. When given more information, ChatGPT performs better on recall for trustworthy labels, and similar in other metrics.

6 Conclusion

Although TOWER performed poorly on the human labelled data, there does seem to be reasons for this which could be rectified to improve its accuracy. The issue may also be due to the skewed nature of the labelling, and so a more evenly distributed human labelling may improve the results for TOWER. Overall, there does seem to be hope of improving TOWER.

For future work, TOWER's ability to test image classifiers could be improved and built on. Currently, it is very early in its development, and is somewhat inaccurate compared to word classifiers, however there is some potentially. There is also the option of using more explanation tools to decrease reliance on LIME. Currently, SHAP is implemented, however it is a lot slower compared to LIME and wasn't used for experimentation. The methods TOWER uses to determine trustworthiness can also be expanded and improved on, with more experimentation required. Getting a less skewed dataset for human labels will also be

helpful.

References

[1] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning testing: Survey, landscapes and horizons," 2019.

[2] B. Li, P. Qi, B. Liu, S. Di, J. Liu, J. Pei, J. Yi, and B. Zhou, "Trustworthy ai: From principles to practices," 2022.

[3] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," 2016.

[4] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," 2017.

[5] A. Groce, T. Kulesza, C. Zhang, S. Shamasunder, M. Burnett, W.-K. Wong, S. Stumpf, S. Das, A. Shinsel, F. Bice, and K. McIntosh, "You are the only possible oracle: Effective test selection for end users of interactive machine learning systems," *IEEE Transactions on Software Engineering*, vol. 40, no. 3, pp. 307–323, 2014.

[6] D. Kaur, S. Uslu, A. Durresi, S. Badve, and M. Dundar, "Trustworthy explainability acceptance: A new metric to measure the trustworthiness of interpretable ai medical diagnostic systems," in *Complex, Intelligent and Software Intensive Systems* (L. Barolli, K. Yim, and T. Enokido, eds.), (Cham), pp. 35–46, Springer International Publishing, 2021.

[7] M. Cheng, S. Nazarian, and P. Bogdan, "There is hope after all: Quantifying opinion and trustworthiness in neural networks," *Frontiers in Artificial Intelligence*, vol. 3, 2020.

[8] H. Jiang, B. Kim, M. Y. Guan, and M. Gupta, "To trust or not to trust a classifier," 2018.

[9] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Mueller, "How to explain individual classification decisions," 2009.

[10] S. Uslu, D. Kaur, S. Rivera, A. Durresi, M. Durresi, and M. Babbar-Sebens, *Trustworthy Acceptance: A New Metric for Trustworthy Artificial Intelligence Used in Decision Making in FoodâEnergyâWater Sectors*, pp. 208–219. 04 2021.

[11] K. Pei, Y. Cao, J. Yang, and S. Jana, "DeepXplore," in *Proceedings of the 26th Symposium on Operating Systems Principles*, ACM, oct 2017.

[12] E. Strumbelj and I. Kononenko, "An efficient explanation of individual classifications using game theory," *J. Mach. Learn. Res.*, vol. 11, p. 1â18, mar 2010.

[13] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, Apr. 2018.

[14] P. S. R. Aditya and M. Pal, "Local interpretable model agnostic shap explanations for machine learning models," 2022.

[15] T. Bolukbasi, K.-W. Chang, J. Zou, V. Saligrama, and A. Kalai, "Man is to computer programmer as woman is to homemaker? debiasing word embeddings," 2016.

[16] R. Schuster, T. Schuster, Y. Meri, and V. Shmatikov, "Humpty dumpty: Controlling word meanings via corpus poisoning," 2020.

[17] S. Kumar, P. Kaur, and A. Gosain, "A comprehensive survey on ensemble methods," in *2022 IEEE 7th International conference for Convergence in Technology (I2CT)*, pp. 1–7, 2022.

[18] Z. Ke and V. Ng, "Automated essay scoring: A survey of the state of the art," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6300–6308, International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[19] 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*, (Online), pp. 4198–4205, Association for Computational Linguistics, July 2020.

[20] S. Jain, S. Wiegreffe, Y. Pinter, and B. C. Wallace, "Learning to faithfully rationalize by construction," 2020.

[21] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," 2018.

[22] Y. Zhang, X. Huang, J. Ma, Z. Li, Z. Luo, Y. Xie, Y. Qin, T. Luo, Y. Li, S. Liu, Y. Guo, and L. Zhang, "Recognize anything: A strong image tagging model," 2023.

[23] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," 2016.

[24] J. e. a. Lehmann, "Dbpedia â a large-scale, multilingual knowledge base extracted from wikipedia," 2015.

[25] E. Saravia, H.-C. T. Liu, Y.-H. Huang, J. Wu, and Y.-S. Chen, "CARER: Contextualized affect representations for emotion recognition," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (Brussels, Belgium), pp. 3687–3697, Association for Computational Linguistics, Oct.-Nov. 2018.

[26] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, (Portland, Oregon, USA), pp. 142–150, Association for Computational Linguistics, June 2011.

[27] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, (Seattle, Washington, USA), pp. 1631–1642, Association for Computational Linguistics, Oct. 2013.

[28] K. Lang, "Newsweeder: Learning to filter netnews," in *Machine Learning Proceedings 1995* (A. Prieditis and S. Russell, eds.), pp. 331–339, San Francisco (CA): Morgan Kaufmann, 1995.