Staying True to Your Word: (How) Can Attention Become Explanation?

Martin Tutek and **Jan Šnajder**
Text Analysis and Knowledge Engineering Lab
Faculty of Electrical Engineering and Computing, University of Zagreb
Unska 3, 10000 Zagreb, Croatia
{martin.tutek,jan.snajder}@fer.hr

Abstract

The attention mechanism has quickly become ubiquitous in NLP. In addition to improving performance of models, attention has been widely used as a glimpse into the inner workings of NLP models. The latter aspect has in the recent years become a common topic of discussion, most notably in work of Jain and Wallace, 2019; Wiegreffe and Pinter, 2019. With the shortcomings of using attention weights as a tool of transparency revealed, the attention mechanism has been stuck in a limbo without concrete proof when and whether it can be used as an explanation. In this paper, we provide an explanation as to why attention has seen rightful critique when used with recurrent networks in sequence classification tasks. We propose a remedy to these issues in the form of a word level objective and our findings give credibility for attention to provide faithful interpretations of recurrent models.

1 Introduction

Not long since its introduction, the attention mechanism (Bahdanau et al., 2014) has become a staple of many NLP models. Apart from enhancing prediction performance of models and starting the trend of fully attentional networks (Vaswani et al., 2017), attention weights have been widely used as a method for interpreting decisions of neural models.

Recently, the validity of interpreting the decision making process of a model through its attention weights came under question. Jain and Wallace (2019) introduced a set of experiments on English language sequence classification tasks which demonstrated that attention weights do not correlate with feature importance measures, and that attention weights generated by a trained model can be substituted and modified without detriment to model performance. While it is natural to assume that multiple plausible explanations for a model's decision can coexist, the authors show the existence of attention distributions that assign most of their mass to words seemingly irrelevant to the task, while still not affecting neither the decision nor the confidence of the model. In the follow-up work, Wiegreffe and Pinter (2019) find that, while such *adversarial* attention distributions do exist, they are seldom converged to in the training process, even when one introduces a training signal with the sole purpose of guiding the model to such distributions.

In this paper, we aim to tackle the difficult question of the relationship between attention and explanation from a different angle – is there any modification we can make to the existing models so that attention could be reliably used as a tool of model transparency? For the sake of consistency, we follow previous work (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019) and limit our scope to single-sequence binary classification tasks, where we consider models from the *RNN + self-attention* family. Concretely, we analyse single-layer bidirectional LSTM-s (Hochreiter and Schmidhuber, 1997) equipped with the additive (Bahdanau et al., 2014) and dot-product (Vaswani et al., 2017) self-attention mechanisms.

Inspired by the recent results (Voita et al., 2019), which show that optimizing the masked language modelling (MLM) (Devlin et al., 2019) objective results in high mutual information between the input and output layers of models, we ask ourselves whether such a trait is beneficial for interpretability. The task of sequence classification in no way incentivizes a model to retain information from the input, and the model is likely to filter out information irrelevant to the task.[1] We believe this lack

[1] The LSTM cell even has an inductive bias towards forgetting information, as we cannot expect the cell gates to always be saturated on the positive side.

Proceedings of the 5th Workshop on Representation Learning for NLP (RepL4NLP-2020), pages 131–142

of enforced information retention causes a discrepancy between the input and hidden vectors, which results in reduced model interpretability. To enforce information retention, we propose a number of techniques to keep the hidden representations closer to their input representations, improving the faithfulness of interpreting models through inspecting their attention weights.

The contributions of this paper are as follows: we (1) investigate whether the lack of a word-level objective causes attention not to be a faithful interpretation, (2) propose various regularization methods in order to improve interpretability through inspecting attention weights, and (3) quantitatively and qualitatively evaluate whether and how these methods help model interpretability.

The rest of the paper is organized as follows. Firstly (§2), we position ourselves within current work and discuss the use of attention as interpretation in NLP,. We then (§3) present our experimental setup, introduce various regularization methods, and briefly describe the experiments we use to evaluate our regularized models. In §4, we offer a quantitative evaluation of the effect of regularizes on the trained models across a number of datasets. We then (§5) qualitatively and quantitatively inspect the effect of regularization on a trained model, identifying what we believe to be the cause of negative results reported in previous work. Finally (§6), we summarize our findings and propose possible lines of future work.

2 Attention and Interpretability in NLP

Preliminaries: Let the input sequence of word embeddings be denoted as $\{w_t\}_{t=1}^{T}$, where T is the length of the sequence. The sequence of hidden states produced by the encoder is then $\{h_t\}_{t=1}^{T}$, where each $h_t = \text{rnn}(x_t, h_{(t-1)})$. The RNN used is a bidirectional LSTM. When discussing a hidden state h_t, we refer only to x_t as its *input* for convenience. The attention mechanism produces a probability distribution over the hidden states, the elements of which we denote $\{\alpha_t\}_{t=1}^{T}$, and refer to as *attention weights*.

2.1 Attention as Interpretation

When interpreting models through the attention mechanism, we assume that the attention weight on the t-th word, α_t, is a *faithful* measure of importance of the input word x_t for the classifier decision. This assumption allows us to *interpret* the decision of the classifier by retrieving the highest attention weights assigned by the model, and then identifying the input words in these timesteps. Thus, in the terminology of Doshi-Velez and Kim (2017), our *cognitive chunk* (a basic unit of explanation) is a single word. However, we are using a BiLSTM as an encoder, and every hidden state is contextualized by virtue of observing the entire input sequence, so the attention weights actually pertain to the input word in context. A faithful measure of importance should by definition accurately represent the true reasoning behind the final decision of the model.[2] So, if attention weights are a faithful measure of importance of word inputs, they will assign large weights to words relevant for the classifier decision.

To define faithfulness more clearly, we can assume the existence of an oracle method which can partition each input sequence of words[3] into *decision-relevant* and *decision-irrelevant* words, where relevance is defined by the judgment of a human reading the text with respect to a task. By this definition, a faithful attention distribution would consistently attribute all or at least most of its probability mass to the decision-relevant words, making it a *plausible* explanation for humans. In contrast, a *counterfactual* attention distribution (Jain and Wallace, 2019) attributes most (or a significant amount) of its probability mass to task-irrelevant words. Obviously, infinitely many plausible and counterfactual explanations exist for a given input instance – merely by redistributing the original attention mass within the same set of words we can obtain infinitely many *alternative* interpretations that are still either plausible or counterfactual.

Jain and Wallace (2019) and Vashishth et al. (2019) demonstrate that, if we permute or substitute the weights of a learned attention distribution, our model can still retain high (and in some cases, unchanged) classification performance and prediction confidence. Even more worryingly, some of the modified attention distributions assign high attention weights to task-irrelevant words while not affecting the instance classification. The existence of such counterfactual attention distributions raises doubts whether inspecting attention weights can be used as a faithful interpretation of the model's decision making process at all.

[2]For an excellent discussion on interpretation faithfulness, see Alon Jacovi's post on https://tinyurl.com/y92rskfr

[3]The instance-level definition is important here, as the same word can bear different meanings in different contexts.

Wiegreffe and Pinter (2019) provide two counter-arguments – (1) *Existence does not entail exclusivity*, suggesting that, just because our model has converged to an attention distribution (a *base* attention distribution), that distribution is not necessarily unique, and alternative attention distributions can still be faithful; (2) while models which produce counterfactual distributions do exist and can be found by post-hoc modifications, these distributions are difficult to converge to naturally through the optimization process of a neural network. This is demonstrated by the authors in experiments where they specifically optimize for a distribution significantly different from the base one.

In contrast, Rudin (2019) states that even if a small fraction of explanations produced by the model is counterfactual, one cannot trust other explanations produced by the same model. Lipton (2016) is more forgiving, and allows that models can still be trusted if they make mistakes, provided humans would also make mistakes on the same instances. The work of Pruthi et al. (2019) emphasizes the threat of interpreting models through attention weights, as they show a regularization term can be introduced to guide the attention weights away from focusing on subsets of words while retaining model accuracy, implying that models which exploit bias in data can be trained to hide the true reasoning behind their decisions.

Among other work, Serrano and Smith (2019) apply an array of tests to analyse whether attention weights correlate with impact on model prediction, concluding again that attention is not a fail-safe (faithful) indicator of importance. The experiments of Vashishth et al. (2019) show that for single-sequence classification, learned attention distributions can be replaced without affecting performance – indicating that attention might not be all we need, after all.

3 Experimental Setup

The **base** model used in (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019) is a single-layer bidirectional LSTM augmented with either a dot-product or an additive attention mechanism, the output of which is then fed into a linear classifier (decoder). We use the same base model as a baseline throughout our experiments.

3.1 Regularizing Models

As mentioned before, we suspect that the lack of a word-level objective weakens the relationship between h_t and x_t, and, consequently, the faithfulness of interpreting attention weights α_t as an explanation of the decision making process of the model diminishes. We will now present a number of methods constructed with the goal of improving information retention between the inputs and hidden states.

Our self-attention augmented LSTM encoder with inputs x_t is defined as:

$$\begin{aligned} e_t &= \mathrm{emb}(x_t) & \alpha_t &= \mathrm{attn}(h_t) \\ h_t &= \mathrm{rnn}(e_t) & s &= \sum \alpha_i h_i \end{aligned} \tag{1}$$

where attn is either the dot-product or additive attention mechanism. The sequence representation s is then fed into a linear decoder.

The simplest way to retain information from input is to include it explicitly in the hidden representations. This can be done by concatenating the embeddings to the hidden representation:

$$h_t^{cat} = [\mathrm{rnn}(e_t); e_t] \tag{2}$$

where $[\cdot;\cdot]$ is the concatenation operator. Another method is to incorporate a residual connection:

$$h_t^{res} = e_t + \mathrm{rnn}(e_t) \tag{3}$$

We use these two methods as our regularized baselines (**concat**, **residual**), along with the unreguralized **base** model.

Our next proposed method is to add a regularization term constraining the L2 norm of the difference between a word embedding and its corresponding hidden representation. As we suspect that the base model discards a lot of word information it deems task-irrelevant, we wish to penalize it for doing so where this information filtering is not crucial.

$$\mathcal{L}_{tying} = \frac{\delta}{T} \sum_{i}^{T} \|h_t - e_t\|_2^2 \tag{4}$$

where δ is the regularization scale hyperparameter, and we minimize the average across all tokens in the batch. We consider values $[1, 10, 20, 30]$ for δ and perform ablation for these values. Further on, we only report results of the model with the best-performing results due to space limitations. We further refer to this method as **tying**.

The last model we propose is inspired by results in (Voita et al., 2019), where we introduce the masked language modelling objective (Devlin et al., 2019), in which input tokens from a sequence are masked at random.[4] The task of the model is then to correctly predict the masked tokens based on contextual cues from the unmasked tokens in the sequence.

In addition to the standard model in (1), the **MLM** model also performs the following:

$$\begin{aligned} \hat{x_t} &= \text{mask}(x_t) \\ \hat{e}_t &= \text{emb}(\hat{x}_t) \\ \hat{h}_t &= \text{rnn}(\hat{e}_t) \end{aligned} \tag{5}$$

The hidden states $\hat{h}_t$ for the corresponding masked tokens are then fed into a linear decoder which predicts the masked word. The encoder and embedding matrix are shared between the MLM and classification tasks.

The MLM linear decoder also introduces no new parameters as we tie the weights (Inan et al., 2016) of the MLM decoder and the input embedding matrix and keep them frozen during training. Both of these choices are motivated by the fact that the model might converge to a solution which does not require retention of information from inputs. In order to apply weight tying, we have to ensure that the dimension of the BiLSTM hidden state equal to the input embedding, and therefore we increase the LSTM hidden state size to 150, compared to 128 in (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019). We also use the new hidden state size for all experiments with the base model.

The MLM setup introduces two hyperparameters: p_{mlm}, denoting the probability of masking a token in a sequence, and η, denoting the weight of the MLM loss. We keep p_{mlm} fixed at 0.15 throughout the experiments, as in (Devlin et al., 2019), and adjust η with respect to the average sequence length in various datasets so that the MLM loss would not dominate the optimization process.[5]

3.2 Post-hoc Modification of Attention Distributions

As suggested by Jain and Wallace (2019), robustness of classifier confidence with respect to attention weight modifications is not a desirable property of interpretable models. Ideally, if a model produces the same decision for an alternative set of attention weights, we would like to be sure that the alternative explanation is faithful. This is not the case in practice as Jain and Wallace (2019) and Vashishth et al. (2019) show that a trained network is surprisingly robust to changes to the attention weights and produces nearly unchanged classification scores even for adversarial distributions. So, while attention is an integral part of training the network, the weights it produces do not greatly affect the classifier decision once trained.

While we agree with the observation of Wiegreffe and Pinter (2019) that robustness of model decisions with respect to attention weights is not necessarily bad as the model is unlikely to naturally converge to such a solution, we believe that fragility of model decisions is an argument **in favor** of interpretability as it indicates that the number of explanations plausible to the model has been reduced, and we perform experiments with that in mind.

3.3 Training an Adversary

In the experiment introduced by Jain and Wallace (2019), for a trained model we attempt to find an adversarial attention distribution which maximizes the Jensen-Shannon divergence (JSD) from the base distribution produced by the trained model, while at the same time minimizing the total variation distance (TVD) from the confidence of the predictions of the base model. The authors demonstrate that it is possible to find an attention distribution that obtains a high JSD while still producing the same prediction confidence consistently across multiple tasks.

As these adversarial distributions were found in an artificial setting, Wiegreffe and Pinter (2019) explore a more realistic scenario and construct an optimization task where, given a fixed (original) model, they train an adversary to minimize TVD from per-instance prediction confidences, while maximizing JSD between per-instance attention distributions of the original model and the adversary. The optimization objective for our adversarial model a given a base model b is defined as follows:

$$\mathcal{L} = \text{TVD}(\hat{y}_a, \hat{y}_b) - \lambda \text{JSD}(\alpha_a, \alpha_b) \tag{6}$$

This training setup introduces another hyperparameter λ, which weighs the JSD component of the

[4] To be precise, either masked, replaced by a random word, or left unchanged. We direct the reader to (Devlin et al., 2019) for a detailed explanation of the MLM task.

[5] As due to keeping p_{mlm} fixed, the longer the sequence is, the more masked predictions we are expected to make.

optimization objective. TVD and JSD are defined as follows:

$$\text{TVD}(\hat{y}_a, \hat{y}_b) = \frac{1}{2}\sum_{i=1}^{|\mathcal{Y}|} |y_{ai} - y_{bi}| \quad (7)$$

$$\text{JSD}(\alpha_a, \alpha_b) = \frac{1}{2}(\text{KL}[\alpha_a||\bar{\alpha}] + \text{KL}[\alpha_b||\bar{\alpha}]) \quad (8)$$

where $\bar{\alpha} = \frac{\alpha_a + \alpha_b}{2}$.

Initially, we were enthusiastic about this setup and conducted the same experiments with our model variants, but drawing any conclusions from the analysis proved to be hard. Firstly, by optimizing for TVD from a trained model instead of on the raw labels, we bias our new model to make the exact same mistakes as the trained model. We believe this severely limits the search space of the adversarial model, as repeating the same mistakes will also bias the model towards exploiting similar patterns in data and, consequently, a similar attention distribution. Secondly, without knowing what the plausible explanations are for the dataset, it is impossible to determine whether a high JSD is a symptom of the model finding an alternative or adversarial explanation. Thus, we do not attempt to draw many conclusions from this experiment, but we reproduce it for completeness with previous work.

3.4 Mutual Information

To quantitatively evaluate whether the regularization has strengthened the relationship between the hidden states and input representations of our model, we look into a recent method of Voita et al. (2019) inspired by the "Information Bottleneck" (IB) theory (Tishby, 1999), where the authors measure an estimate of mutual information (MI). Originally applied to transformers (Devlin et al., 2018), this method is straightforward to adapt to the bidirectional LSTM.

Similarly to our point of view, the IB theory states that neural networks, in general, aim to extract a compressed representation of input in which information relevant for the output is retained while irrelevant is discarded. Mutual information is used as a method of measuring how much information is lost between the input and hidden representation of a certain network. Voita et al. (2019) show transformer networks discard progressively more information in deeper layers. This phenomenon is different for the case of MLM in transformers, where MI is higher in the uppermost layers, likely due to the task of reconstructing corresponding input tokens.

The strength of the relationship between e_t and h_t can be quantified by estimating MI. As MI is intractable to compute in the continuous form, we first discretize the vector representations and estimate MI in the discrete form. Following Voita et al. (2019) and Sajjadi et al. (2018), we perform this discretization by clustering the embedding and hidden representations to a large number of clusters and using the obtained cluster labels in place of the continuous vectors to estimate MI.

Concretely, we select a subset of 1000 words from the vocabulary and gather at most 1M representations of these tokens at input and hidden level. We then cluster the obtained representations into $k = 1000$ clusters with mini-batch $k-$means with batch size of 100. We obtain the vocabulary sample in two ways: as the top 1k most frequent words (**MF**), as in (Voita et al., 2019), but also as a random sample (**RS**) of from the scaled unigram distribution.[6]

3.5 Datasets

We experiment on the following English language datasets for binary classification tasks, which were either originally built for this task or were adapted for it by Jain and Wallace (2019):

(1) *The Stanford Sentiment Treebank (SST)* (Socher et al., 2013), a collection of sentences tagged with sentiment on a discrete scale from 1 to 5, where 1 is the most negative and 5 the most positive. We omit the neutral class (3) and conflate scores 1 and 2 as well as 4 and 5 into negative and positive class, respectively;

(2) *IMDB Large Movie Reviews Corpus* (IMDB) (Maas et al., 2011), a binary sentiment classification dataset of movie reviews;

(3) *AG News Corpus*, a categorized set of news articles from various sources. We limit ourselves to binary classification between articles labelled as *world* (0) and *business* (1);

(4) *20 Newsgroups* similarly, we consider the task of discriminating between *baseball* (0) and *hockey* (1) in this dataset of newsgroup correspondences labelled with 20 categories;

(5,6) *MIMIC ICD9* (Johnson et al., 2016), a dataset of patient discharge summaries from a database of electronic health records. Here, we

[6]The sample is drawn from the unigram distribution raised to the power of $\frac{3}{4}$.

analyse two classification tasks on different subsets of the data: whether a summary is labelled with the ICD9 code for *diabetes* (1) or *not* (0) (henceforth *Diabetes*) and whether a summary corresponds to a patient with *acute* (0) or *chronic* anemie (henceforth *Anemia*);

For consistency, we use the train/test/dev splits produced by Jain and Wallace (2019).[7]

4 Results

4.1 Attention is Fragile

We report the average F1-scores of five runs for the **base** model and the following regularization variants: **concat**, **tying**, and **MLM**. We omit results on **residual** due to space, but they are consistently comparable to **concat** due to their similar nature. For each model variant we report results of experiments with the dot-product (•) and additive (+) attention mechanism. Due to space constraints, we omit the full results and refer the reader to Appendix for more details.

We report the performance of each model in scenarios where we use trained attention (**Tr.**), a random permutation of the trained attention (**Pm.**) or substitute the attention distribution with the uniform (**Un.**). For the uniform and permutation settings, we report the drop in F1-score when compared to trained attention performance.

We omit the results on the Diabetes dataset, as every modification of attention weights on this dataset results with an F1-score of 0, due to a very small number of tokens being a high-precision indicator of the positive class, as noted by Jain and Wallace (2019). As shown in Table 1, regularization setups increase fragility of model performance with respect to modifications of the attention distribution, while retaining similar classification scores to the base model. These results indicate that we have successfully reduced the space of possible alternative explanations for the model by tying the input and hidden representations closer together. By doing this, we show that lateral information leakage (between hidden states) is reduced when proper regularization is applied, and that, as a consequence, alternative explanations are also plausible. Having shown this, we still need to determine whether a high attention weight on a hidden state is a faithful measure of importance of a corresponding input.

[7] https://github.com/successar/AttentionExplanation

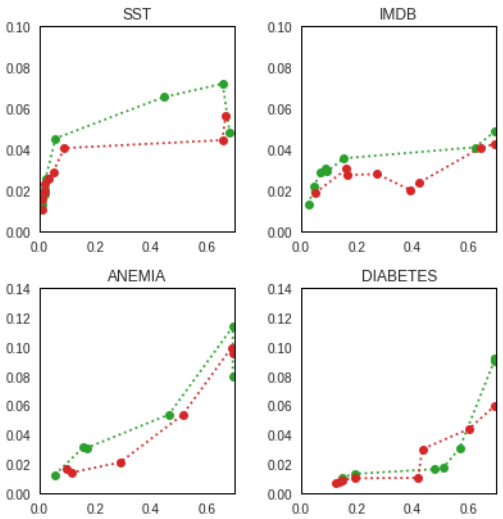

Figure 1: Averaged per-instance test set JSD (x-axis) and TVD (y-axis)

4.2 Mutual Information is Higher

In Table 3 we report mutual information scores across datasets for the most frequent words (MF) and a random sample drawn from the scaled unigram distribution of the vocabulary (RS).

The increase in mutual information scores between inputs x_t and hidden states h_t implies that more information from the inputs is retained during encoding. While retention of input information is not a desirable trait of a model performing pure sequence classification, as the only goal the model optimizes is producing the correct class label with high confidence, it is beneficial for interpretability. If we wish to interpret classifier decisions through inspecting attention weights on hidden states, we have to ensure that a hidden state preserves a significant degree of information from the input. A significant increase in mutual information suggests that the base model was filtering or overwriting a large amount of information from the input, making attention inspection less credible. It is not possible to report mutual information for the **concat** setup as the dimensionality of the hidden vector is larger than the input embedding, so we report the results for **Residual**. The results for the Residual setup can be considered close to the best realistically obtainable MI score as the model explicitly includes the input embedding in the hidden state.

4.3 Adversarial Attention Distributions are Harder to Find

In Fig. 1 we report results where for a fixed oracle model we train an adversary with the objective

		Base			Concat			Tying			MLM		
	α	↑ Tr.	↓ Un.	↓ Pm.	↑ Tr.	↓ Un.	↓ Pm.	↑ Tr.	↓ Un.	↓ Pm.	↑ Tr.	↓ Un.	↓ Pm.
SST	+	**84.2**	−2.8	−5.0	83.7	−1.9	−4.7	83.5	**−7.0**	**−18.0**	82.8	−5.9	−15.6
	•	**84.3**	−2.6	−6.5	84.1	−3.4	−7.4	83.5	**−9.9**	**−20.0**	82.7	−3.0	−5.4
AG	+	**95.9**	−2.3	−3.9	**95.9**	−1.5	−2.6	95.0	**−3.3**	**−14.6**	95.2	−1.5	−6.6
	•	95.9	−2.3	−3.8	**96.1**	−1.9	−3.0	95.4	**−3.0**	**−12.2**	95.4	−2.0	−5.3
NG	+	90.9	−9.8	−14.1	91.3	−25.0	−28.2	91.4	−39.7	−43.2	**91.5**	**−76.3**	**−66.0**
	•	**91.1**	−35.2	−36.8	91.0	−40.4	−37.1	90.9	−37.0	−42.8	89.1	**−79.6**	**−72.8**
IM	+	**88.3**	−10.0	−13.4	**88.3**	−10.2	−14.0	87.1	**−56.2**	**−43.3**	87.5	−22.8	−26.5
	•	**88.2**	−18.6	−22.9	87.9	−17.2	−20.8	87.2	**−57.7**	**−45.3**	87.8	−15.3	−18.5
ANM	+	92.4	−21.6	−22.4	**92.8**	−19.3	−22.2	91.3	−31.4	−27.6	89.7	**−35.0**	**−37.7**
	•	**92.7**	−10.2	−14.4	92.4	−15.2	−17.2	91.0	**−91.0**	**−59.7**	90.7	−37.8	−33.9

Table 1: % F1-scores for trained models (higher is better) and drops in performance (Δ F1) when applying regularization (lower is better). Scores reported are averages over five runs.

	$\sim$	Base	Resid	Tying	MLM
SST	MF	2.324	**5.062**	4.870	3.662
	RS	2.435	**4.289**	4.216	3.808
AG	MF	1.940	**5.467**	4.075	3.845
	RS	2.078	**4.518**	4.177	3.980
NG	MF	1.566	**4.345**	3.985	3.677
	RS	1.828	**3.843**	3.784	3.458
IM	MF	2.455	4.998	**5.186**	3.728
	RS	2.682	**4.366**	4.434	3.885
ANM	MF	3.711	**5.253**	4.239	4.016
	RS	3.780	**4.477**	3.950	3.921

Table 2: Mutual information scores between the input and hidden representations. Higher is better. Due to space limitations, results are only reported on additive attention.

of minimizing the TVD between the predictions of the model and, at the same time, maximizing JSD between per-instance averaged attention distributions. Due to space limitations, we only report results for the MLM regularised model, while the others fare comparably. The red dotted line indicates the imitation setup of the base model, and the green dotted line indicates imitation setup for the MLM model. Consistently, except for an outlier point in the Diabetes dataset, the imitation setup of the MLM model produces larger drops of TVD in order to increase the JSD between attention distributions, corroborating the claim that attention distribution of the MLM model is more fragile.

5 Understanding the Effect of Model Regularization

To visually demonstrate the undesired effect of attention mechanisms when trained in the **base** setting, as well as to illustrate the effect of regularizations we applied, we first analyse how we obtain the classifier prediction. The output of the classifier is an affine transformation of the attention output:

$$\begin{aligned} p_{logit} &= W_d(\sum_{i=1}^{T} \alpha_i h_i) + b_d \\ &= \sum_{i=1}^{T} \alpha_i (W_d h_i + b_d) \\ &= \sum_{i=1}^{T} \alpha_i \hat{p}_t \end{aligned} \tag{9}$$

We can reformulate this as a convex attention-weighted sum of logits ($\hat{p}_t$) obtained by running each individual hidden state through the decoder. Once we scale the logits for individual timesteps, we obtain the prediction probability as if the whole attention mass was on that hidden representation. For attention weights to be a faithful measure of interpretability, this probability should be high only on tokens which are decision-relevant.

In Fig. 2, we plot these token-level probabilities for a single example to demonstrate that in the **base** model, this is not the case. We can see that for the base model, the probabilities for most tokens have nearly the **same** probability as the final prediction, while the regularization keeps the representations for neutral words grounded closer to the decision boundary. As a direct result of this, the model predictions are much more fragile to change of attention weights, as only a small number of hidden states are far enough from the decision boundary to produce an equally confident classification.

We now quantitatively formulate and measure this criterion – **if** the accuracy of a regularized classifier isn't hurt by the regularization, when optimizing for interpretability we should prefer models that have a lower per-token average prediction probability (given that the prediction for that instance is correct).

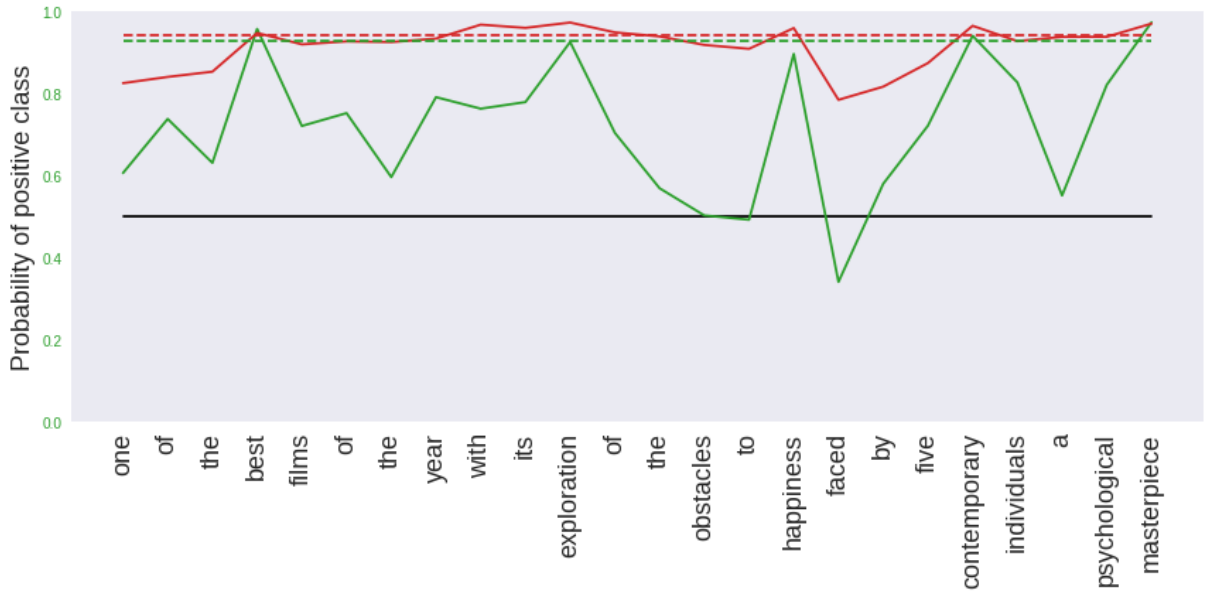

Figure 2: Per-token prediction probability for an example from the SST dataset for the base model (red) and a regularized (tying) model (green). The dotted lines indicate the classification probability of the model. More instances and examples of other regularization techniques can be found in the Appendix.

		Base	Resid	Tying	MLM
SST	+	0.712	0.685	**0.586**	0.630
	•	0.693	0.701	**0.600**	0.664
AG	+	0.887	0.822	**0.615**	0.695
	•	0.862	0.876	**0.646**	0.698
NG	+	0.811	0.551	0.577	**0.514**
	•	0.687	0.755	0.516	**0.482**
IM	+	0.625	0.609	**0.533**	0.562
	•	0.590	0.608	**0.539**	0.553
AN	+	0.568	0.547	0.531	**0.515**
	•	0.542	0.534	0.515	**0.519**

Table 3: Average per-token prediction probability across models and tasks. From the perspective of interpretability, lower is better, given the classifier performance is not significantly affected.

6 Conclusion

We have identified the lack of a word-level objective as the likely cause of attention weights not being a faithful tool of interpretability in the case of sequence classification with attention mechanism augmented recurrent networks. We experimentally establish that we can add regularization methods to sequence classification which strengthen the relationship between the input and hidden states while not being a detriment to classification performance. If one wishes to interpret classifier decisions through inspecting attention weights, we strongly suggest inclusion of a technique such as weight tying or adding masked language modelling as an auxiliary. Adding such methods causes the model to become more susceptible to attacks modifying the attention weights of a trained model, and increases faithfulness of explanations produced by attention weights.

While we believe our work is a step forward towards using attention weights as a faithful explanation, by no means do we claim that the modification is sufficient. As was our primary concern, the risk with using attention weights as a tool of interpretability is that a single bad explanation could have consequences in decision-making scenarios, and while our methods improve the faithfulness of such interpretability, it is by no means foolproof. We have only scratched the surface of faithful interpretability, and most of the datasets in our and previous work do not have human annotated rationales. In order to fully understand the cases in which attention provides a reliable explanation, we believe that datasets with annotated rationales or decision-relevant tokens should be used. This analysis should also be extended to more complex models which better capture the nuances of language. We believe that the experiments we presented demonstrate the shortcomings of interpreting model decisions through inspecting attention weights, however we acknowledge that this branch of research sorely lacks evaluation methods that include humans in the loop.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2016. Tying word vectors and word classifiers: A loss framework for language modeling. *arXiv preprint arXiv:1611.01462*.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035.

Zachary C Lipton. 2016. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C Lipton. 2019. Learning to deceive with attention-based explanations. *arXiv preprint arXiv:1909.07913*.

Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. 2018. Assessing generative models via precision and recall. In *Advances in Neural Information Processing Systems*, pages 5228–5237.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

N Tishby. 1999. The information bottleneck method. In *Proc. 37th Annual Allerton Conference on Communications, Control and Computing, 1999*, pages 368–377.

Shikhar Vashishth, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. 2019. Attention interpretability across NLP tasks. *arXiv preprint arXiv:1909.11218*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. 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)*, pages 4387–4397.

Sarah Wiegreffe and Yuval Pinter. 2019. 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)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

General parameters	
Embedding dim	300
RNN hidden dim	150
Learning rate	1e−3
Grad. clipping	5
Batch size	32
Weight decay	1e−5
Regularization parameters	
Masking prob.	0.15
Masking weight η	$\{0.1, 0.3, 1, 3, 5\}$
Tying weight δ	$\{10, 20, 30\}$

Table 4: Model hyperparameters

A Model Hyperparameters

Since we analyse a number of models and regularization techniques, we naturally also have a large number of hyperparameters. We do not tune any of them except for regularization-specific ones and we inherit others them from previous work (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019). A notable change is the dimension of the hidden state, which we increase from 128 to 150 due to the nature of the MLM regularization. We, however, repeat the experiments for the base model with this increased dimensionality.

We report our parameters in Table 4. While we have considered other values in a brief search for η and δ, but we have only ablated over the mentioned ones as they have proven to be (locally) optimal.

Dataset	Avg. len.	Vocabulary
SST	17	17310
AG News	31	15286
20NG	164	15590
IMDB	234	41919
Diabetes	1700	23778
Anemia	1927	20290

Table 5: Statistics of datasets used in experiments

We also report the statistics of datasets used in experiments in Table 5. The average instance length had a significant impact on the experiments as datasets with longer instances were naturally more fragile to attention distribution modifications.

B Experiments on Multilayer LSTMs

All of the experiments performed in the paper used single-layer LSTMs. Even though the considered binary classification tasks could be considered some of the simplest NLP problems, one still wonders what would the effect be if a more complex encoder was used. To this end, we perform a preliminary set of experiments where we use the best hyperparameters used for training of the single-layer networks and increase the number of layers of the LSTM network.

The results in Table 6, while far from conclusive, show that (1) among all tasks, the base model consistently becomes **more robust** to attention perturbation the more layers we add. Inconsistently, we further observe a (2) **diminishing return** of regularization techniques among tasks as the number of layers increases. In some cases, the 3-layer results do not follow this trend (but, curiously, the regularization seems to have a stronger effect). We believe that these results should be taken with a grain of salt prior to a careful ablation study, but still might interest the reader.

C Importance of Initialisation in Dot-Product Attention

Initially, the experiments we conducted worked well for additive attention but not for scaled dot-product attention. While the various regularization techniques produced significant changes in F1-scores when the additive attention distribution was modified post-hoc, this was not the case for dot-product attention and the F1-scores remained constant no matter the modification. This was caused by the fact that the attention distribution of the model consistently converged to a uniform one.

After exhaustive experimenting, the only change that fixed this behavior was changing the default initialization scheme for the query parameter. The dot-product self-attention mechanism for a **single instance** (for illustrative purposes) is generally defined as follows:

$$\text{Attention}(q, K, V) = \text{softmax}(\frac{qK^T}{\sqrt{d_k}})V \quad (10)$$

where q is the query vector, while K and V are stacked representations for each timestep. In practice, when using self-attention for single-sequence classification, the query is a model parameter,[8] while the keys and values are functions of RNN hidden states. In our case concretely (following Jain and Wallace (2019); Wiegreffe and Pinter (2019)),

[8]This independence of the query vector from the instance is not intuitive in our perspective (it seems natural to us that different information is relevant for different instances), but in practice we find that both approaches work equally well.

		Base			Concat			Tying			MLM		
	#L	↑ Tr.	↓ Un.	↓ Pm.	↑ Tr.	↓ Un.	↓ Pm.	↑ Tr.	↓ Un.	↓ Pm.	↑ Tr.	↓ Un.	↓ Pm.
SST	1	**84.2**	−2.8	−5.0	83.7	−1.9	−4.7	83.5	**−7.0**	**−18.0**	82.8	−5.9	−15.6
	2	84.2	−1.0	−1.2	**84.5**	−0.8	−4.6	84.1	−3.7	**−14.5**	84.4	**−3.8**	−5.9
	3	84.2	−0.7	−0.7	83.3	−1.3	−1.3	**84.6**	−2.7	−13.9	82.3	**−7.3**	**−18.0**
AG	1	**95.9**	−2.3	−3.9	**95.9**	−1.5	−2.6	95.0	**−3.3**	**−14.6**	95.2	−1.5	−6.6
	2	95.7	−0.3	−0.3	**95.9**	−1.4	−2.0	95.5	**−3.7**	**−14.8**	95.6	−1.6	−3.8
	3	**95.9**	−0.0	−0.1	95.7	−1.0	−1.6	95.4	−2.0	−12.8	95.8	**−13.2**	**−62.5**
NG	1	90.9	−9.8	−14.1	91.3	−25.0	−28.2	91.4	−39.7	−43.2	**91.5**	**−76.3**	**−66.0**
	2	93.7	−0.9	−5.6	**94.0**	−6.3	−11.9	92.8	−17.5	−25.5	89.8	**−31.6**	**−35.0**
	3	**92.0**	0.0	0.0	91.5	−30.2	−35.7	89.0	**−30.3**	**−39.3**	88.5	−17.9	−17.4
IM	1	**88.3**	−10.0	−13.4	**88.3**	−10.2	−14.0	87.1	**−56.2**	**−43.3**	87.5	−22.8	−26.5
	2	88.4	−3.1	−3.8	**88.9**	−7.2	−9.1	87.6	**−51.2**	**−41.1**	87.4	−14.5	−21.7
	3	88.5	−1.2	−1.4	**88.8**	−5.7	−7.8	87.9	−7.6	−21.3	87.1	**−87.1**	**−84.5**

Table 6: % F1-scores for trained models (higher is better) and drops in performance (Δ F1) for LSTM models with multiple layers. The number of layers is indicated in the second column.

the keys and values are the hidden states themselves.

With this in mind, Eq. 10 can be written as follows:

$$\text{Attention}(H) = \text{softmax}(\frac{L_q(H)}{\sqrt{d_k}})H \qquad (11)$$

where L_q is the trainable query parameter. In our `Pytorch` implementation, L_q is a `Linear` layer, which is initialised from the Kaiming uniform[9] distribution with the scale parameter $\sqrt{5}$. With this initialisation, the dot-product attention distribution in our experiments has always converged to a uniform one. When we changed the initialisation to instead sample from a standard normal distribution, the dot-product attention converges to a sensible distribution. We suspect this problem occurs because the small initial weights of the linear transform scale down the difference norm between the attention probabilities too much to be distinguished from the uniform distribution.

D Additional Visualisations of Regularization Effects

To expand on Fig. 2, we now plot per-token prediction probabilities for multiple models. We sometimes omit the model classification probabilities not to clutter the plots too much. We select diverse examples (Figs. 3–7) from the first three batches of the SST validation split.

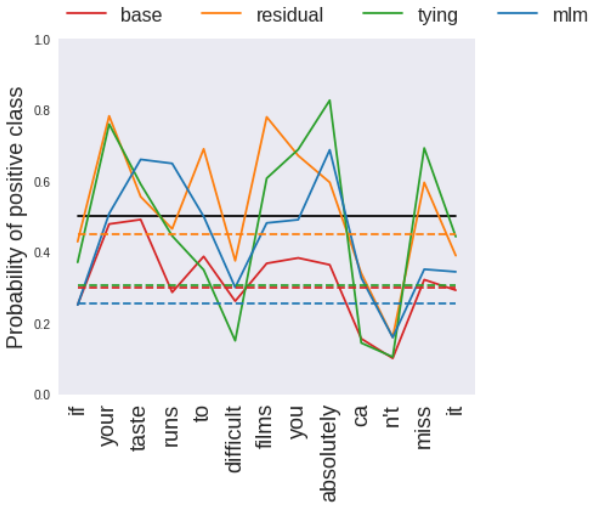

Figure 3: A negative example: perhaps the analysed single-layer LSTM is unable to understand even the simple nuances of language. Here the instance is classified as negative across all models only due to presence of the word "difficult". Note that these models obtain a near 0.9 F1-score on this dataset.

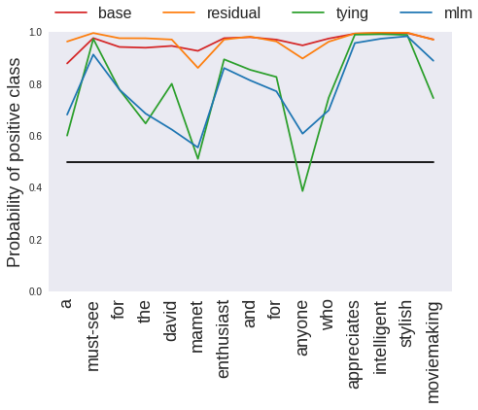

Figure 4: A clear-cut instance

[9] https://github.com/pytorch/pytorch/blob/master/torch/nn/modules/linear.py#L79

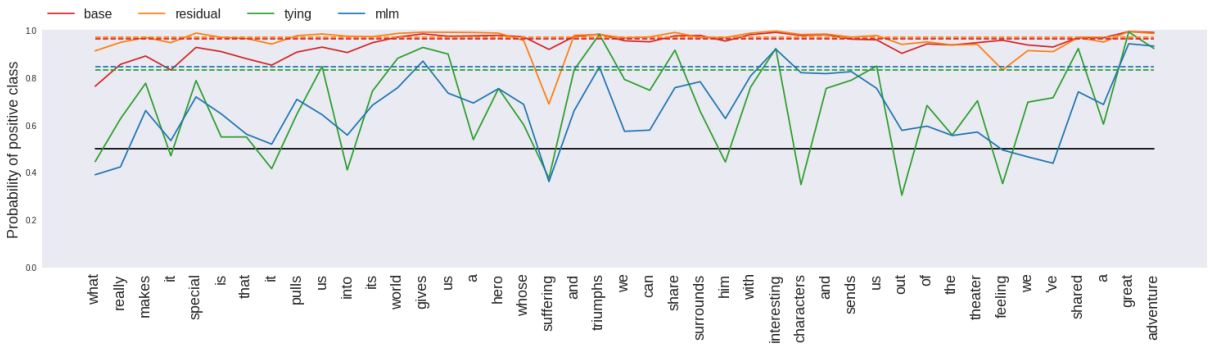

Figure 5: A long example which further demonstrates lateral information leakage

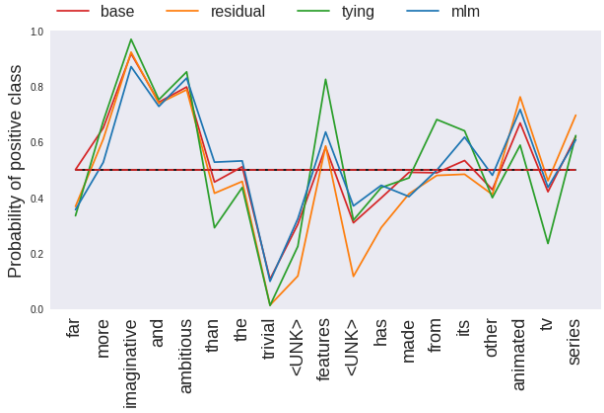

Figure 6: We observe that for instances where the model is not clear about the classification, the per-word probabilities are pretty similar between regularizations. We believe that lateral information leakage happens only when the model is confident in its prediction. Base model prediction confidence is indicated in this example (it overlaps with the 0.5 line).

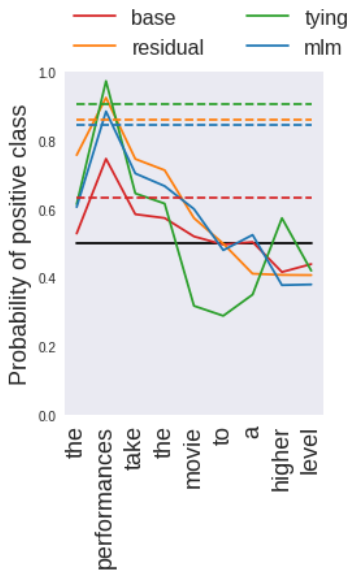

Figure 7: A rare example where the regularised models are more confident in the correct prediction than the base model