Cohesive Attention-Based Explanations for Sequences and Explainability in Presence of Event Types

Stephanie Ger[1], Yegna Subramanian Jambunath[2], Diego Klabjan [3], Jean Utke [4]

[1] Department of Engineering Sciences and Applied Mathematics, Northwestern University, Evanston, Illinois, USA
[2] Center for Deep Learning, Northwestern University, Evanston, Illinois, USA
[3] Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois, USA
[4] Allstate Insurance Company
July 13, 2023

Abstract

While many methods such as Locally Interpretable Model-agnostic Explanation (LIME), Integrated Gradients and Layerwise Relevance Propagation (LRP) have been developed to explain how recurrent neural networks make predictions, the explanations generated by each method often times vary dramatically. There is no consensus about which explainability method most accurately and robustly determine features important for model prediction. We consider a classification task on a sequence of events with different types and apply both gradient-based and attention-based explanation models to compute explanations on the event type level. We show that attention-based models return a higher similarity score between explanations for models initialized with different random seeds. However, there are still significant differences in explanations between model runs. We develop an optimization-based model to find a low-loss, high-accuracy path between two sets of trained weights to understand how model explanations morph between different local minima. We use this low-loss path to provide insight as to why explanations vary on two sentiment datasets.

1 Introduction

In diverse industries spanning from healthcare to finance, there are interesting problems that can be investigated through the analysis of time series data. With the advent of recurrent neural networks (RNNs), many classification and text generation tasks have been tackled with Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRUs) [7,12]. The effectiveness of recurrent neural networks for both text generation tasks and sequence classification tasks such as sentiment analysis has led to many improvements on the underlying recurrent neural network units [13]. For example, time aware LSTMs (T-LSTMs) have been developed in order to deal with the case when time gap between events in a sequence differ [6].

While work has been done to improve model performance for RNNs, less has been done to understand how these models are making predictions. In particular, it is often quite difficult to determine which inputs cause the model to make a certain prediction. This is why neural networks as a whole have long been referred to as "black box" models because unlike decision trees or logistic regression, the model architecture does

not provide a easy path toward explaining a given prediction. Work has been done on the explainability of neural networks, but there is no consensus on which explainability method performs best.

For many tasks, the fact that neural network models outperform standard classification techniques is sufficient to justify the use of neural network models. However, there are applications where it is important to understand why the model makes a prediction. For example, when it comes to medical data, we want to know if the model is capturing features that are known to be important to a given diagnosis [22]. Explanations can also be used to identify model failure; e.g. in [24], a classifier was trained to determine if emails were about atheism or Christianity, but when an explainability model was applied to the classifier, it was clear that the classifier was looking at differences in email headers and not the content.

While work has been done to tackle explainability of neural networks both as a whole and specifically for recurrent neural networks, models for temporal data generally focus on how a given feature impacts the model prediction. However, if we instead need to use codes to make a medical diagnosis, then we are interested in how each medical code affects model prediction. Attention-based mechanisms have been applied to these problems, but if the same medical code, or event type, occurs multiple times in a given sequence, then it is unclear how these existing attention mechanisms should be used to understand the importance of each event type [5, 8, 30]. In such a context explanations should not be position dependent since they rely only on event types which are position independent.

We encounter similar issues when gradient-based methods are used to examine event-level features for tasks such as the sentiment analysis. For classification model training, an embedding such as BERT, Doc2Vec or Word2Vec [9, 21] is generally used for the words, so when gradients are computed with respect to these embedded features, it is unclear how the gradients for each feature should be combined into a single value for each event (word, token). As with attention-based explainability, gradient values must be consolidated in order to get an explanation for each event type.

We develop an attention-based method that returns a single attention value for each event type even if multiple events of the same type occur in the same sequence. First, attention is used to compute a context vector for each event. Then a context vector is computed across all event types with respect to the event level context vectors and the corresponding attention vectors are returned as the explanation vector. A single architecture is initialized with different initial seeds and the same optimization method is used to update model weights. We observe that the model accuracy and loss metrics are similar across a range of different initial seeds. However, when explanations are clustered and evaluated using a clustering comparison score such as the Adjusted Rand Index, the same model architecture trained with the same optimization routine returns substantially different explanation vectors for each initial seed. On the Amazon review dataset, we created three models by using three seeds with each model having similar performance on test. Table 1 reveals that the resulting explanations based on attention are drastically different. (ARI of 1 means identical clusters). This unexplained phenomenon is not due to explainability of the event types. Several experiments on a variety of standard datasets find the same inconsistency when explanations are done at the token (event) level.

Run	**0**	**1**	**2**
0	1	0.25	0.31
1	0.25	1	0.20
2	0.31	0.20	1

Table 1: ARI Scores for Attention Explainability

We examine this issue by developing an optimization based low-loss trail algorithm to compute a low-loss path between two sets of trained weights. The straight line interpolation path is taken between the two sets of weights until the next step would exceed the maximum allowable loss; at that point a constrained optimization problem is solved to determine a direction along which the loss decreases but does not deviate too much from the straight line. This path allows us to investigate how model explanations morph between trained model weights on such a low-loss, high accuracy path in order to understand why the explanation vector clustering scores differ across seeds.

The main contributions of this paper are as follows.

1. A hierarchical attention model to compute a single explanation for each event type.
2. An experimental study of the following explainability models for sequences:
   (a) Hierarchical Attention
   (b) Attention
   (c) Gradient-based explainability
   (d) Integrated Gradients

   As part of the study, we employ explanation vector output clustering for each explanation model and we provide a comparison of clustering across models with different initialization.
3. An optimization based low-loss trail algorithm to determine a low-loss path between two sets of trained weights.
4. Insights on how model explanations morph between trained weights based on the low-loss path between two sets of weights.

In the next section, we discuss relevant literature. The proposed explanation models and the low-loss trail algorithm are discussed in Section 3. The comparison of explanation methods and the results of the optimization based low-loss trail algorithm are presented in Sections 4 and 5, respectively. Section 6 discusses how model explanations morph between two sets of trained weights.

2 Literature Review

One of the main drawbacks of recurrent neural networks is that it is unclear how a model makes the prediction. Many different techniques have been developed to explain how certain inputs affect the model

prediction by examining the effect of each feature. General explainability models that can be applied to RNNs include both sensitivity analysis based models such as Layerwise Relevance Propagation (LRP) and DeepLIFT as well as models such as Locally Intepretable Model-agnostic Explanations (LIME) [3, 10, 24]. On the other hand, sequence specific explanation models include attention-based models as well as modifications to the model architecture so that explanations can be computed with respect to each class [11, 22].

Sensitivity analysis methods consider the gradient of the output with respect to each of the inputs to determine how sensitive each of the inputs are to change. By determining how sensitive the model is to changes in the input, we can get a sense of the local shape of the decision space [10]. However, these methods are not robust to the use of activation functions such as ReLUs or zero inputs due to non-differentiability. Methods such as integrated gradients [29], DeepLIFT [28], and LRP [2] improve on sensitivity analysis techniques by increasing the robustness of the explanation. LRP does so by determining the impact, positive or negative, of each input on the prediction of each possible classification label using techniques from backpropagation to compute the importance of each input to the layer, called the relevance score, for each layer. The equations used to assign the relevance score depend on the network architecture and must satisfy a conservation principle [2]. LRP has been used to explain both image classification and sentiment analysis models [3, 4].

On the other hand, both DeepLIFT and Integrated Gradients seek to improve on sensitivity analysis by comparing a given input x to a baseline input x'. For image data, an appropriate baseline could be a black image, that is a zero image, while for text data an appropriate baseline could be a zero embedding vector or the embedding for a stop word. The precise rationale for chosing x' is problem dependent. DeepLIFT does so by using LRP type calculations in a way that utilizes both x and x' while Integrated Gradients does so by computing the line integral from the given input to the baseline input [28, 29]. In all attribution methods, we want to satisfy the principle of completeness [29], which states that the difference between the classification model prediction for x and x' should equal the total attributions determined by the explainability model. As it can be shown that the DeepLIFT attribution equations do not satisfy the completeness principle for recurrent neural networks due to the gating mechanisms in the RNN units [1], they cannot be used to compute a meaningful explanation for RNNs. Therefore, we benchmark our proposed model against both the Integrated Gradients and the sensitivity analysis models.

In contrast to sensitivity analysis based methods, LIME aims to explain a model output by both determining an explainable version of the input vector and a linear model that approximates the model output of the deep model [24]. These approximations provide a local explanation of the model prediction and can be used to determine how certain inputs affect model prediction. LIME has been used to explain both image classification models as well as LSTM sentiment analysis models [24, 25]. The downside to LIME is that the scope of the local explanation is unclear as we do not know how far away from a given input the explanation is valid. An extension to LIME, the Anchor method, has been developed as a way to determine a set of rules, e.g. the presence or absence of certain features, for the prediction of each target class [25]. These rules are developed by determining which features have high coverage, that is, occur in many of the samples in a given distribution, and paring down this set of features until the desired precision on the rule set is achieved. As with LIME, SHapley Additive exPlanations (SHAP) is a local explainability model that uses a game theoretic approach to assign explanations to each feature that is present for a given sample [18]. While these models can be applied to any model, because an explainable model must be trained for each sample, it is computationally expensive to run on multiple samples. This makes running LIME on the entire dataset to understand which inputs are important to each class prediction prohibitively expensive.

Another class of explainability models are models that aim to explain how predictions are made by looking at certain components of the model. One such model decomposes the LSTM equations in order

to determine the contribution made by a given subset of events to the model prediction [23]. Attention mechanisms have been used in sequence to sequence models to improve model performance by allowing the model to focus on the most relevant time steps of the encoder output [5]. As attention provides insight as to which features are most relevant to the decoder at each time point, the attention weights can be used to assign an importance to each event in the time series. Attention-based explainability methods have been employed in various settings [11, 22], however, none of these models take event type into account which would necessitate aggregations at position level explanations driven by event types. In our approach we fuse explanations at positions corresponding to the same even type (this is considered for each different event type). It is possible to have multiple events of the same event type in a sequence, and when that happens it is unclear how to combine these attention values into a single attention value for each event type. Hierarchical attention has been used to compute attention at different levels of granularity for sentiment analysis tasks [31], but models have not been applied to this problem of duplicate features in the input. We use this idea of hierarchical attention to develop a model that computes a single attention for each feature in the dataset regardless of multiplicity. We benchmark this proposed model against a standard attention-based explanation model.

In order to understand the explanations generated by these attribution methods, work has been done to visualize these model explanations [26]. For image data, this is straightforward as explanations generated from sensitivity analysis type methods can be plotted and visualized to understand which parts of the image are relevant for model prediction [1]. On the other hand, it is less clear how to visualize the explanations for sequence type data. Work on the visualization of sequence data has focused on looking at the activations of individual LSTM neurons [14]. While this work has yielded interesting results, especially with respect to sentiment analysis problems, there does not appear to be a systematic approach to examining cell activations. In addition, if the input is not text, but instead another type of multi-feature temporal data, it is more difficult to draw conclusions about what information each LSTM cell is encoding. Currently there are no methods to quantify model explanations for temporal neural network models.

3 Approaches

We assume that the data is of the form $x^i = (e^i_1, e^i_2, \ldots, e^i_T)$, $e^i_t \in \mathbb{R}^n$, $y^i = (y^i_1, y^i_2, \ldots, y^i_K)$, $y^i_k \in \mathbb{R}^m$, where y^i is the label sequence for the sample feature sequence x^i. The input data is a variable length sequence of events or tokens where each event is encoded as a feature vector, using an embedding method such as word2vec or BERT [9,21]. Each event e^i_j has its type $t(e^i_j)$. When dependency on sample i is not needed, we omit the superscript. The set of all unique event types, $\{t(e^i_j)\}_{i,j}$, in the dataset is for simplicity written as $\{ET_1, \ldots, ET_Q\}$. In a given sequence it is possible there to be multiple events with the same event type. Our main intent is in designating explainability at the event type level and to explore a way to reconcile different explanations from different models with similar performance. We first overview existing explainability models for sequences that are used in the computational study. Then we propose our explainability model directly addressing event types. In the end, we discuss a path-based method addressing inconsistency in explanations.

3.1 Existing Explainability Models

One of the downsides of existing explanation models is that once an explanation is computed for each event in the sequence, the explanations must then be aggregated by event type. For all three existing explanation methods that we consider, the explanations for each event are aggregated (typically by summation) by type.

In contrast, the hierarchical attention model presented later computes a single explanation value for each event type directly.

A sequence to sequence model consists of an encoder and a decoder where the output of the former provides input to the latter. For the attention sequence to sequence model, once the classification model is trained, we can use the attention weights for the trained model to determine each feature's importance to the model prediction. Given a standard attention mechanism where $h^E = [h_1^E, \ldots, h_T^E]$ is the encoder hidden state output (h_i^E is the state vector between two consecutive neural network embedding cells), we have the context vector

$$\bar{h}_{\bar{t}}^E = \sum_{t=1}^{T} \alpha_{t\bar{t}} h_t^E, \alpha_{t\bar{t}} = \text{softmax}\left(V \cdot \sigma(A h_t^E + B h_{\bar{t}-1}^D)\right), \bar{t} = 1, \ldots, K$$

where V, A, B are trainable weights and $h_{\bar{t}}^D$ are decoder hidden states. Here $\bar{h}_{\bar{t}}^E$ represents the attention vector that is an additional input to cell at position $\bar{t}$ in the decoder. The context vector is used as the input to the decoder LSTM and the attention weights $\vec{\alpha}_{\bar{t}}$ used to weight the encoder hidden states are used as the explanations for each event in the sequence.

We cannot compute gradient-based explanations directly from the model weights unlike the attention-based model. Instead, given output of a trained classification model M on sample x represented in the same form as x^i, we compute the gradient with respect to each feature $e_{tj}, e_t = (e_{tj})$, i.e. for each position t and feature j we have

$$g_{tj} = \frac{\partial}{\partial e_{tj}} M(x), t = 1, \ldots, T, j = 1, \ldots, n$$

Unlike attention explanations, the gradients $\sum_j g_{tj}$ must be summed to get an explanation for each event in the sequence.

Finally we consider the Integrated Gradients method, which is an extension to standard gradient-based explanations. In this model, the gradients computed along the straight line path from a baseline input x' to the input sequence x^i are summed to determine an explanation of which input features are most important for the classification. As with the sensitivity analysis, the explanations for each feature are summed to get an explanation for each event in the sequence.

3.2 Hierarchical Attention

While we are able to compute an explanation vector for event types by brute force using standard explainability methods, we would like to compute an explanation vector directly from the attention values without any ad-hoc additional aggregation methods (summations or average are possible aggregation techniques but they are in essence arbitrary choices). To do so, we introduce a Hierarchical Attention method which gives us a way to determine a single attention value for each event type during model training. As before we denote by h^D and h^E standard hidden states between cells. For each event, t, in the sequence such that $e_t = ET_j$ and at each decoding step, $\bar{t}$, we compute the following quantities

$$f_{t\bar{t}} = V \cdot \sigma(A_1 h_t^E + B_1 h_{\bar{t}-1}^D)$$
$$r_{j\bar{t}} = \text{softmax}_{i=ET_j}(f_{i\bar{t}})$$

in order to compute a context vector

$$h_{j\bar{t}} = \sum_{i=ET_j} r_{j\bar{t}i} h_i^E$$

for each event type indexed by $j = 1, \ldots, Q$. Now, we can use the event type context vectors to compute a context vector for the entire encoder sequence from

$$\begin{aligned} d_{j\bar{t}} &= U \cdot \sigma\left(A_2 h_{j\bar{t}} + B_2 h^D_{\bar{t}-1}\right) \\ \alpha_{j\bar{t}} &= \text{softmax}(d_{j\bar{t}}) \\ \bar{h}^E_{\bar{t}} &= \sum_{j=1}^{Q} \alpha_{j\bar{t}} h_{j\bar{t}}. \end{aligned}$$

With the context vector, $\bar{h}^E_{\bar{t}}$, and the previous decoder hidden state, we write the decoder hidden state update equation as

$$h^D_{\bar{t}} = f\left(h^D_{\bar{t}-1}, c_{\bar{t}-1}, y_{\bar{t}-1}, \bar{h}^E_{\bar{t}}\right)$$

where f is the standard LSTM equation and U, V, A_i and B_i are all trainable weights. In the Hierarchical Attention model, we have a single attention value $\alpha_{j\bar{t}}$ for each event type j, so we can use the attention vector directly as the explanation vector.

3.3 Low-Loss Trail Algorithm

In the introduction we give cases when different initial seeds lead to different explainability vectors. We would like to understand how the explainability vectors for different models are related. It is known that given a sufficiently over-parameterized model and two sets of trained weights, we can find a path between the two sets of weights, $\vec{w}_{path}$, such that the loss for all weights $w \in \vec{w}_{path}$ is under a certain threshold [15]. In [15], they prove that this low-loss path exists using a dropout based approach where it is hard to check the assumptions and the algorithm itself requires modification of matrices which is hard to implement. For these reasons, we consider an alternative optimization based approach. To this end, we develop an optimization-based algorithm to determine this path in order to examine how the explanations evolve between trained weights of different models.

Given loss function ℓ and two models with weights w^1, w^2, we want to find a sequence of weights $\hat{w}_1 = w^1, \hat{w}_2, ..., \hat{w}_M = w^2$ such that

$$\ell(\hat{w}_i) \leq L_{\max} = \max(\ell(w^1), \ell(w^2)) + \varepsilon \tag{1}$$

and

$$||\hat{w}_i - \hat{w}_{i-1}|| \leq J \cdot ||w^2 - \hat{w}_{i-1}|| \tag{2}$$

for $i = 2, 3, ..., M$. Here J and ε are two hyperparameters. In (2) we constrain the relative distance to the target; other versions are possible. To determine such a path (see Figure 1 and Figure 2), we first take steps along the straight line path between the weights, until we reach a point, $\hat{w}$, where the maximum allowable loss L_{max} is reached. The weights generated are at most J apart. At this point we can no longer continue on the direct path and thus we have to take a "detour." We need to find a direction d that decreases the loss and also does not deviate much from the straight line.

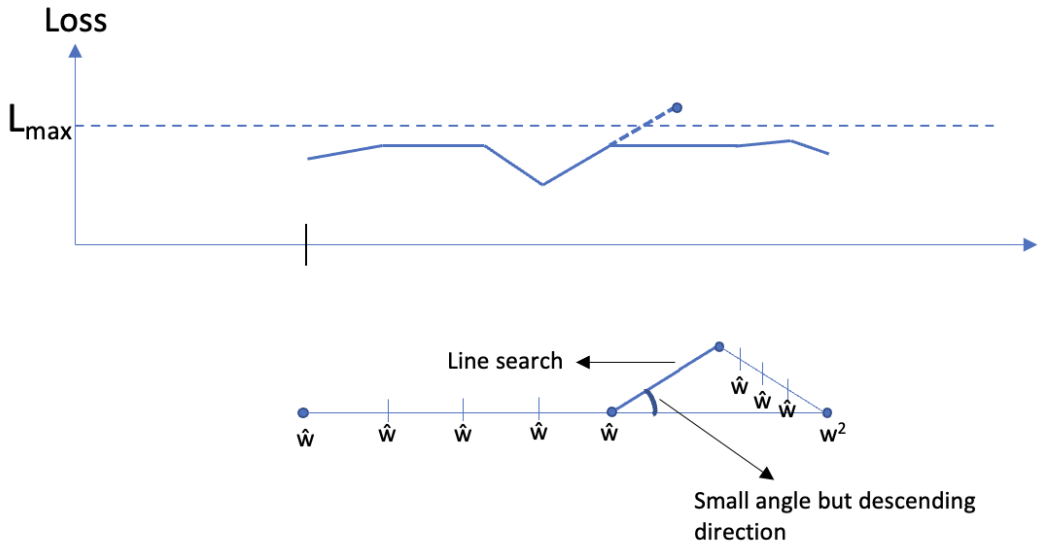


Figure 1: Loss Trajectory of the Algorithm

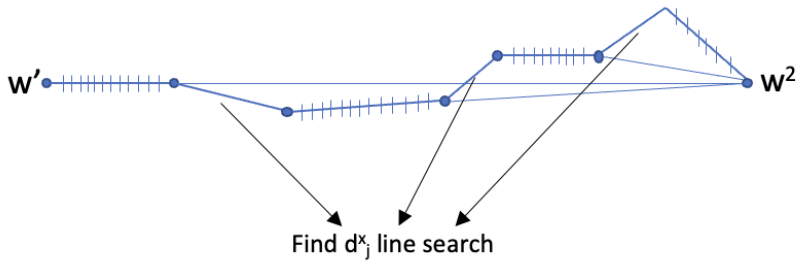


Figure 2: Representative Trajectory of Weights if there are only 2 of them

To determine a new direction to take from the point $\hat{w}$, we solve the optimization problem

$$Dir(\hat{w}) = \min_{d} \frac{-(w^2 - \hat{w})^T d}{||w^2 - \hat{w}||||d||}$$

subject to the constraint $\nabla \ell(\hat{w})^T d \leq 0$. This allows us to determine a direction, d, where the angle between d and the straight line path $w_2 - \hat{w}$ from the current iterate $\hat{w}$ is minimized and the loss in direction d decreases. We use a Lagrangian subgradient method to solve this constrained optimization problem. Once we have the optimized direction, d^*, we use a linesearch method to get $\hat{w}_{new}$. The line search follows Algorithm 1 and obeys (2). After we have taken this step, we take steps along the straight line path between $\hat{w}_{new}$ and the target w^2 until either the target is reached or we pick a new direction d^*. In other words, we repeat the process sequentially with $\hat{w}_{new}$. These steps are detailed in Algorithm 1. We note that constraints (1) and (2) might be infeasible for a given model. Algorithm 1 always satisfies (2) but it dynamically increases the initial ε when no further progress is possible.

Once this path is computed, we examine how the explainability vector changes along this path between trained weights from the different models. We consider how the explainability attributions of the most im-

portant event types change along this path to determine which explanation vector features remain consistent and which features vary. With this path, we can explain how we can have two sets of trained weights, both returning a similar test F1-score, with differing explanations.

4 Explainability Computational Study

We consider three multi-feature temporal datasets where each event in the sequence has an associated event type [1]. We consider two sentiment analysis datasets. One is a dataset that classifies IMDB movie reviews as positive or negative [19]. The other is a dataset that classifies Amazon reviews as positive or negative [20]. For both of these datasets, we use wordnet to generate hypernyms for each word in the dataset, which we then use as the event type [17]. Using hypernyms yields around two thousand event types. Hypernyms are more likely to occur in multiple reviews, making it easier to draw conclusions about the importance of each event type. On both sentiment datasets, we would like to determine which hypernyms are most important for the sentiment analysis task. Finally, a business use case is discussed.

Each dataset contains variable length sequences. Instead of padding the data to the maximum sequence length and then masking the explanation vector, we instead bin samples by sequence length and train on batches of equal length sequences. For each dataset, a sequence-to-one model is trained with either attention or hierarchical attention. Both models are implemented in Keras and trained on a single GPU card. As all tasks are classification tasks, we report the F1-score for each dataset. We compute the explanation vector using the weights associated with the best validation F1-score for each of the following methods: attention, hierarchical attention, gradient, and integrated gradients. The model trained with hierarchical attention is used to compute the hierarchical attention explanation vector and the model trained with attention is used to compute the attention and gradient-based explanations. For each of the datasets, we train models with three different seeds for both the hierarchical attention model and the attention model. We then compute the explanation vector for each of the methods and study the stability of the clustering methods by computing the ARI between the different model runs [27]. This method quantifies the overlap between two separate clustering methods using the contingency table which counts the number of samples in cluster A_i and B_j where A_i and B_j are two clusters based on two clustering solutions A and B. The score increases if samples in cluster A_i are all in B_j.

For the attention, gradient and integrated gradient explanation models, we set the explanation value to 0 for event types that are not present in the sample. For the hierarchical attention model, explanation values are set to 0 for missing event types in the explanation vector method so no additional imputation is necessary. In addition, for the integrated gradients explanation model, different baseline inputs are used for the various datasets. We use the BERT 'OOV' token as the baseline for the sentiment datasets and the zero embedding vector as the baseline for the remaining dataset [9].

We use K-Means clustering to cluster sequences in order to determine if we can derive distinct subsets of the data and to examine each cluster to get a better understanding of the model prediction. In order to determine the number of clusters, K, the silhouette score is computed to visualize how many clusters are appropriate. In order to compute the similarities between the explanation vectors across models initialized with different seeds, we use the ARI.

[1] Code and data are available at `https://github.com/stephanieger/sequence-explainability/`

4.1 Amazon Polarity

We consider all reviews under 100 words long and consider batches of data such that all samples in the same batch are of the same length. We then use a Google pretrained BERT model to embed the dataset [9]. The dataset comprises around 100 thousand samples with 20% in test and there are 2,602 event types.

Run	Attention Model	Hierarchical Attention Model
0	88.7%	87.0%
1	88.1%	87.1%
2	88.1%	88.3%
Average	88.3 %	87.5%
Standard Deviation	0.3%	0.6%

Table 2: Amazon Polarity Test F1-Scores for Each Seed

We see in Table 2 that the F1-scores for each of the model runs are very similar, which suggests that the model results are reproducible. We also note that the hierarchical attention classification accuracy is similar to the standard attention model which shows that if the hierarchical model can provide more similar clustering results and thus explainability across the runs, it is a viable alternative to the standard attention model. We consider clustering results for the standard attention, hierarchical attention, and gradient methods. In Appendix B.1, the number of clusters chosen for each explanation method and the ARI scores for each pair of runs are discussed. These scores are summarized in Table 3.

Model	Mean	Standard Deviation
Attention	0.253	0.045
Hierarchical Attention	**0.373**	**0.025**
Gradient	0.167	0.236
Integrated Gradient	0.334	0.472

Table 3: Amazon Polarity ARI Score Summary

We note that on average the ARI score for the hierarchical attention clusterings on the different model runs is about 10% higher than the ARI score for the attention clustering methods, suggesting that the clusterings for the hierarchical attention models are more stable. To get a better understanding of ARI scores, we examine the clusterings for the pairs of model runs that return the highest and lowest ARI scores. We look at the percent of samples that are assigned to the same clusters by both models. For the attention model, between 72.5% and 77.8% of validation samples are assigned to the same cluster. On the other hand, for the hierarchical attention model, between 83.3% and 84.2% of samples are assigned to the same cluster. We match clusters by solving the maximum assignment problem using the Hungarian method [16]. This implies that the hierarchical attention explainability method is more stable than the standard attention explainability method. We observe that while the average integrated gradient ARI score is high, the standard deviation of the ARI score is high as well for both gradient methods. This suggests that the gradient-based methods does not return as consistent clustering results as either of the attention methods.

4.2 IMDB Sentiment

We consider all reviews under 600 words long and batch data such that all samples in the same batch are of the same length. The Google pretrained BERT model is used to embed the dataset [9]. The dataset comprises around 50 thousand samples with 20% in test and there are 2,538 event types.

Run	Attention Model	Hierarchical Attention Model
0	86.3%	86.1%
1	86.3%	86.5%
2	86.7%	86.0%
Average	86.4%	86.2 %
Standard Deviation	0.2%	0.2%

Table 4: IMDB Test F1-Scores for Each Seed

As with the Amazon polarity dataset, we observe that the F1-scores for each of the model runs are very similar in Table 4. We also note that the hierarchical attention performs very similarly to the standard attention model, which again suggests that we do not lose classification performance in order to obtain more stable explainability results. Appendix B.2 provides details while Table 5 is a summary.

Model	Mean	Standard Deviation
Attention	**0.220**	**0.014**
Hierarchical Attention	0.177	0.026
Gradient	0.097	0.137
Integrated Gradient	10^{-4}	10^{-4}

Table 5: IMDB ARI Score Summary

We observe that the ARI scores for the hierarchical attention and standard attention models are fairly similar. However, unlike the Amazon dataset, the standard attention model returns a higher average ARI score and both attention models return relatively low standard deviations. The difference between the ARI scores for the two different models is lower for the IMDB dataset than the Amazon dataset. It is interesting to note that the difference in ARI scores between the two attention explainability models is smaller for the harder classification task. In addition to returning lower test F1-scores than the Amazon polarity classification models, the ARI scores for the IMDB models are lower than the Amazon polarity models. As we noted in the Amazon polarity model, both gradient-based models return low average ARI scores across model runs.

4.3 Business Use Case

This dataset consists of a prediction task on a sequential dataset. It comes from our industry partner and details are proprietary. The set has 100 thousand sequences and the maximum length is 100. It is a very unbalanced dataset. While there is more variation between the runs, we see in Table 6 that as in the previous two datasets, the hierarchical attention performs slightly worse than the standard attention models. We do not expect the hierarchical attention model to outperform the standard attention model, but we want to

ensure that the difference in model performance is not substantial. We observe in Table 6 that the average test F1-score over 3 runs is 20.5% and 19.9% for the standard averaging and hierarchical averaging models, respectively. The average results are in Table 7.

Run	Attention Model	Hierarchical Attention Model
0	20.6%	19.3%
1	21.0%	20.2%
2	19.8%	20.3%
Average	20.5%	19.9%
Standard Deviation	0.5%	0.4%

Table 6: Test F1-Scores for Each Seed

Model	Mean	Standard Deviation
Attention	**0.200**	0.199
Hierarchical Attention	0.170	**0.042**
Gradient	0.110	0.156
Integrated Gradient	0.194	0.273

Table 7: ARI Score Summary

When we compare pairwise comparisons for the standard attention models, between 33.3% and 66.7% of samples are in the same cluster, while between 58.8% and 67.2% of samples are in the same cluster for the hierarchical attention model. This suggests that the hierarchical attention model explainability vectors are more stable. We observe that while the ARI scores for the standard attention models are on average higher than the ARI scores for the hierarchical attention model, the scores are also less consistent. As we observed in the other datasets, from the standard deviations for the two gradient-based methods, it is clear that the gradient-based models underperform the attention-based methods and are unsuitable for computing event type based explanations.

5 Explainability Interpolation Computational Study

On all three datasets, we observe that attention-based methods outperform gradient-based methods based on the ARI scores between different pairs of runs. We run the low-loss trail algorithm on each dataset to compute the low-loss path between each pair of trained models. This algorithm is implemented in Tensorflow and all paths are computed using a single GPU. The path of weights is computed between the weights that return the highest validation F1-score for both of the two models. The parameter values vary for each dataset and are discussed in each section.

To validate that the algorithm produces a low-loss path between the two trained models, we compare the minimum loss along the path for the low-loss trail path and the straight line interpolation path. We also consider the minimum validation F1-score along both paths as we expect the validation F1-score along the low-loss path to be significantly higher than the validation F1-score along the straight line path. Once the low-loss path has been determined between each pair of runs, the attention or hierarchical attention

explanation vector is computed along the low-loss path for each set of weights. Each explanation vector is then clustered and changes in explanations are examined separately for each cluster. Furthermore, positive and negative samples are examined separately and only correctly classified samples are considered.

5.1 Amazon

We see from Figure 3, that the maximum loss for the low-loss trail path is much lower than the loss along the straight line interpolation path for hierarchical attention between two seeds. The slight jaggedness to the trail loss plot is expected because it shows that when the next step would otherwise result in a loss that exceeds the maximum allowable loss, the optimization-based method selects a descending direction that decreases the model loss. The loss increases along the straight line interpolation after a direction change until either a new descending direction is chosen or the target is reached. This suggests that the optimization based interpolation method is performing as expected.

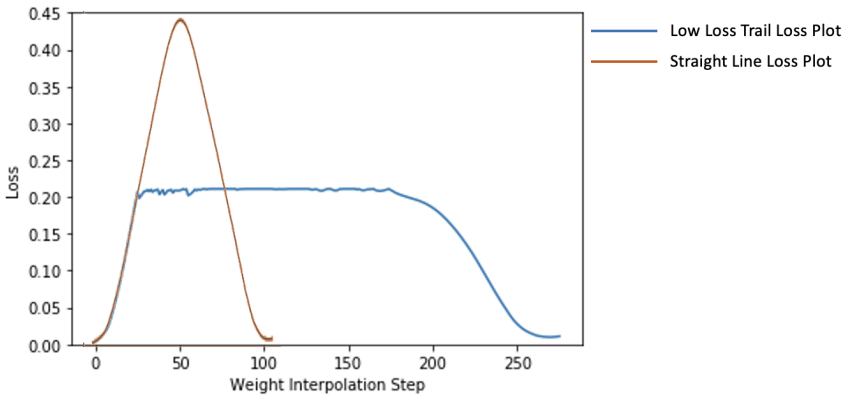


Figure 3: Amazon Loss Along Path

We also consider the minimum loss along both the straight line and low-loss trail paths for the other interpolation model runs and observe in Table 8 that the results are consistent with respect to each pair of model endpoints.

Seeds	Standard Attention		Hierarchical Attention	
	Straight Line	Weight Interp	Straight Line	Weight Interp
0 - 1	0.44	0.21	0.48	0.27
0 - 2	0.49	0.23	0.48	0.27
1 - 2	0.35	0.23	0.44	0.26
Average	0.43	0.22	0.47	0.27

Table 8: Amazon Maximum Loss Along Paths

Table 9 shows that on average the F1-scores for the straight line paths are 8-10% lower than the F1-scores for the low-loss trail path. For the hierarchical attention model, this difference is statistically significant with p-value at 0.02. We compare these minimum F1-scores to the F1-scores for the trained endpoints in Table

10 to observe that the minimum F1-scores along the low-loss trail path are within 5% of the trained weights for both models. This suggests that the path determined by the interpolation algorithm is a low-loss, high-accuracy path and that it is reasonable to consider the explanations along this path.

Seeds	Standard Attention		Hierarchical Attention	
	Straight Line	Weight Interp	Straight Line	Weight Interp
0 - 1	0.77	0.82	0.78	0.83
0 - 2	0.71	0.81	0.72	0.83
1 - 2	0.80	0.82	0.76	0.82
Average	0.76	0.82	0.75	0.83

Table 9: Amazon Minimum F1-Scores Along Paths

Seeds	Standard Attention		Hierarchical Attention	
	Loss	F1-Score	Loss	F1-Score
0	0.01	0.86	0.02	0.87
1	0.01	0.86	0.01	0.87
2	0.03	0.86	0.01	0.87
Average	0.02	0.86	0.01	0.87

Table 10: Amazon Trained Loss and F1-Scores

5.2 IMDB

As we observe in the Amazon polarity dataset, the loss along the low-loss path is substantially lower than the loss along the straight line path between two sets of trained weights as shown in Figure 4. We note that this result is consistent across all interpolation paths in Table 11.

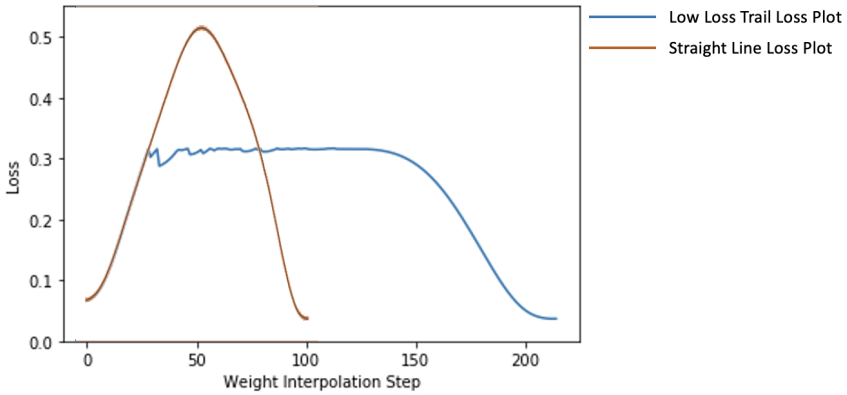


Figure 4: IMDB Loss Along Path

Seeds	Standard Attention		Hierarchical Attention	
	Straight Line	Weight Interp	Straight Line	Weight Interp
0 - 1	0.51	0.32	0.65	0.37
0 - 2	0.64	0.37	0.67	0.39
1 - 2	0.52	0.35	0.54	0.38
Average	0.56	0.35	0.62	0.38

Table 11: IMDB Maximum Loss Along Paths

In Table 12, we observe that on average, the F1-score for the straight line paths are over 13% lower than the minimum F1-score along the low-loss path. Again, we see that this difference is significant for the hierarchical attention model with p-value at 0.04, but not for the standard attention model. When we compare the minimum F1-scores along the low-loss path, we observe that the F1 score is within 6% of the trained weight F1-score for both models. This suggests that we can consider the explanations along the interpolation path as the model performance remains consistent along this path.

Seeds	Standard Attention		Hierarchical Attention	
	Straight Line	Weight Interp	Straight Line	Weight Interp
0 - 1	0.75	0.82	0.72	0.80
0 - 2	0.60	0.80	0.62	0.79
1 - 2	0.76	0.82	0.73	0.80
Average	0.70	0.81	0.69	0.80

Table 12: IMDB Minimum F1-Scores Along Paths

Seeds	Standard Attention		Hierarchical Attention	
	Loss	F1-Score	Loss	F1-Score
0	0.07	0.85	0.07	0.84
1	0.07	0.85	0.07	0.84
2	0.04	0.87	0.03	0.84
Average	0.02	0.86	0.06	0.84

Table 13: IMDB Trained Loss and F1-Scores

5.3 Business Use Case

As we observed in the sentiment datasets, the loss along the low-loss trail path is substantially lower than the loss along the straight line path between the two sets of trained weights as shown in Figure 5. This result is consistent across all interpolation paths in Table 14.

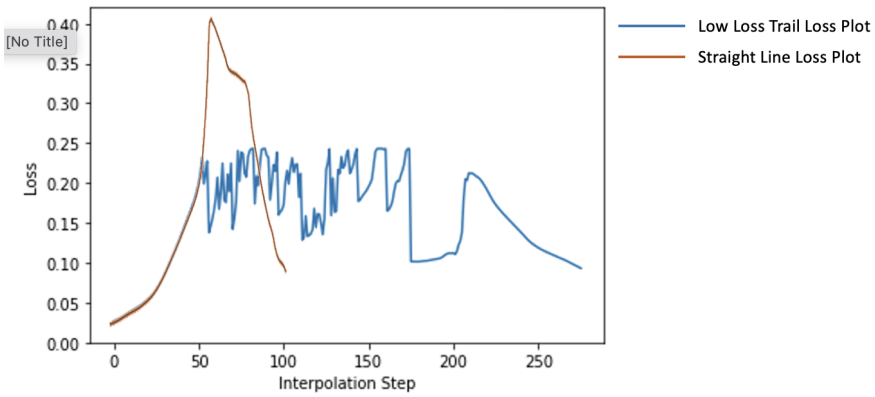


Figure 5: Loss Along Weight Paths

Seeds	Standard Attention		Hierarchical Attention	
	Straight Line	Weight Interp	Straight Line	Weight Interp
0 - 1	0.41	0.24	2.81	0.33
0 - 2	0.80	0.42	1.33	0.26
1 - 2	0.48	0.29	0.86	0.12
Average	0.56	0.32	1.67	0.24

Table 14: Maximum Loss Along Paths

Table 15 shows that despite the decrease in loss observed in Table 14, the F1-scores for the low-loss trail path and the straight line interpolation are fairly similar. A possible explanation is that for this dataset, the F1-score is much more dependent on loss than for the IMDB and Amazon datasets, so a similar decrease in loss does not prevent a decrease in the F1-score.

Seeds	Standard Attention		Hierarchical Attention	
	Straight Line	Weight Interp	Straight Line	Weight Interp
0 - 1	0.10	0.11	0.08	0.08
0 - 2	0.09	0.10	0.13	0.13
1 - 2	0.09	0.11	0.08	0.11
Average	0.09	0.11	0.11	0.11

Table 15: Minimum F1-Scores Along Paths

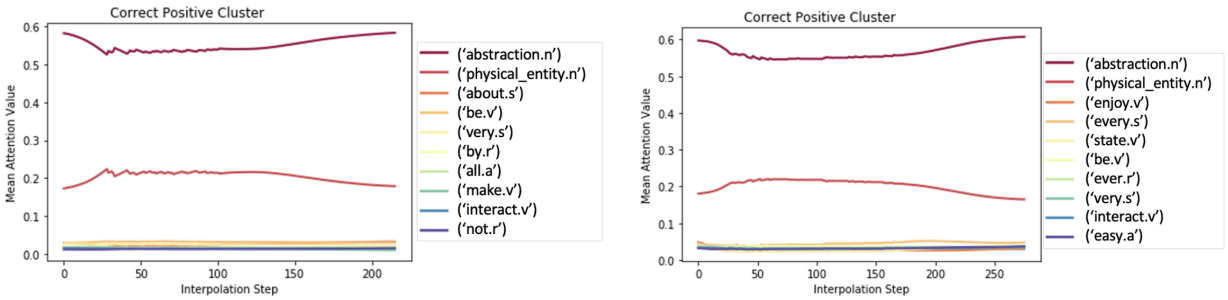
6 Interpretation of Explanation Along Low-Loss Path

In this section, we examine the interpolation plots for the sentiment analysis datasets and draw conclusions about the high attention event types (we cannot repeat the same analysis on the business use case due to

Seeds	Standard Attention		Hierarchical Attention	
	Loss	F1-Score	Loss	F1-Score
0	0.03	0.21	0.01	0.20
1	0.09	0.16	0.01	0.20
2	0.09	0.20	0.00	0.21
Average	0.07	0.19	0.01	0.20

Table 16: Trained Loss and F1-Scores

confidentiality). The high attention event types along the low-loss trail path can be used to determine if there are any trends in the attention which can help explain how explanations for trained endpoints are different. We plot the high attention events separately for event types that occur in 15% of the samples and event types that occur in less than 2% of the samples. We observe that stable event types, ones that are present in a large percent of samples, have attention that is fairly consistent across the interpolation path, while rare event types, ones that occur in a small percent of samples, have variable attention between endpoints.

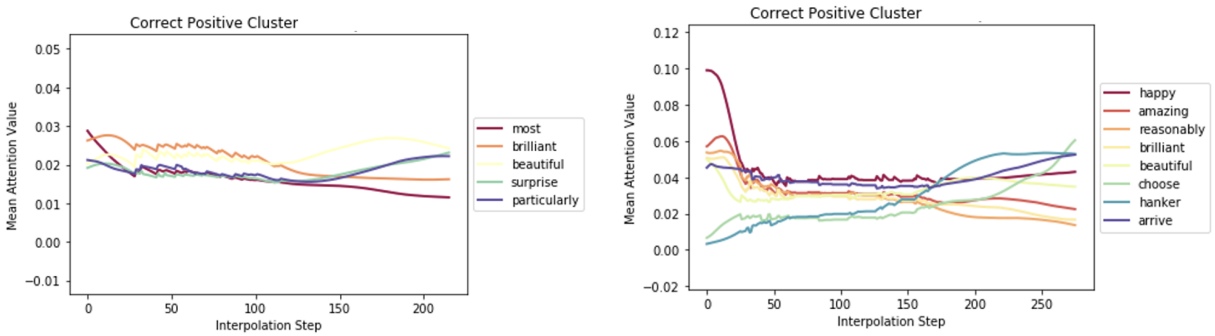


(a) IMDB Stable Event Types

(b) Amazon Stable Event Types

Figure 6: Visualization of Common Event Types for Sentiment Datasets.

When considering the event types that occur in more than 15% of the samples in a given cluster, Figure 6 shows that the attention plots are consistent across the interpolation steps and when comparing the stable event types from one trained model to the other, we observe that the attention for each event type remains stable along the path and the relative order remains consistent. This suggests that the majority of the variability in explanations between trained models is not due to the stable event types. We also observe that stable event types such as 'abstraction' and 'physical_entity' represent a large set of nouns and adjectives, respectively. These are sets of words that can be important to all reviews.



(a) IMDB Rare Event Types

(b) Amazon Rare Event Types

Figure 7: Visualization of Rare Event Types for Sentiment Datasets.

On the other hand, when we examine events that occur in less than 2% of the samples in a given cluster, we observe a substantially different behavior along the low-loss trail path in Figure 7. There are crossing events where we see that some event types are important to one trained model, and others are important to the other model. The rare event types that are important to each endpoint often have high polarity, since event types such as 'brilliant,' 'beautiful,' 'happy,' and 'amazing' are all words that signify positive sentiment in a review. We observe in Figure 6 that a large percent of the attention is dominated by 'abstraction' and 'physical_entity,' so the average attention for the rare event types along the low-loss trail path is lower than the attention for the stable event types. Since the attentions for these event types fluctuate more than the attentions for the stable event types, it appears that differences in rare event type attention lead to the differences in attention clustering for each of the trained models.

We next examine individual reviews for both the IMDB and Amazon datasets to understand the differences in explanation as a result of variations in the attention of rare event types. For each trained model we compute the list of rare event types and consider all reviews with rare event types from both trained models and examine how the rare event type is used. We observe that an event type can either be context dependent, where the sentiment of the event type depends on the neighboring words, or have high polarity, where the event type and neighboring words contain a high sentiment event type. Across both datasets, we consider up to two words before and after the rare event type when determining if a rare event is context dependent or high polarity, unless the rare events from the different endpoints are adjacent or the rare event is at the end of a phrase, in which case we do not consider adjacent words. In the following reviews, we observe examples of rare event type switching events, where the rare event for one model is context dependent and the rare event for the other model has high polarity. The bolded words in each of the phrases below are rare events from different model runs. We observe that "great" has high polarity, while "hidden" is context dependent in the IMDB review. Similarly, we observe in the Amazon review that "happy" has high polarity and "sturdy" depends on the words around it for meaning and is thus context dependent.

> "No **hidden** agenda Pure scifi All fun I saw the original on TV and was scared pretty bad I was a kid The original one can be appreciated more when compared to the new one which I saw and have forgotten The original one starring the **great** movie star Steve McQueen BULLET is by far the better and only version anyone should see The movie production is dated but the fx used to make the Blob stands up the test of time I was convinced that that thing was moving on its own accord 10 10 Zafoid" (IMDB)

"Cool product Keeps my eighteen months old warm and **happy** can cover up to her face convenient in windy situations One small flaw would be the length of the elastic which attaches to the stroller it s too short and threatens to rip anytime It looks **sturdy** enough to resist a little while though It s minor inconvenience Overall I love this product JJ Cole Urban Bundle Me Toddler Soho" (Amazon)

Datasets	Amazon		IMDB	
	Standard Attention	Hierarchical Attention	Standard Attention	Hierarchical Attention
Switching	31	15	50	23
Non-Switching	12	17	80	16
Total	43	32	130	39

Table 17: Count of Switching and Non-switching Events for Samples with Rare Events from both Trained Models.

When we analyze all reviews that contain rare events from both endpoints and consider if a switching event occurs, Table 17 shows that the relative number of switching events differs between the standard attention and hierarchical attention models for both datasets. The ARI scores in Tables 20a, 20b, 18a, and 18b (in the Appendix) show that the there are more switching events when the ARI score is lower. We use the Fisher Exact Test to determine that the increase in switching behavior is significant for both the Amazon and IMDB datasets with a p-value of 0.024 and 0.036, respectively. This suggests that this rare event type switching behavior can partially explain the differences in explanations between models trained with different random seeds.

From the discussion of rare event types, we observe that neighboring words are crucial to understand the behavior of rare events. Therefore, it makes sense to consider the attention of words adjacent to rare events. However, we can only consider attention for individual events for the standard attention method as the hierarchical attention model computes attention for each event type, not for each event in the sequence. When we consider the attention of events in a 5-gram centered at the rare event type, we observe that the maximum attention of the surrounding words in the 5-gram is on average 9% higher than the attention for the rare event itself. This suggests that context is important for understanding sentiment explanations, but that event type based attention is not designed for a context dependent approach. Future work could include the development of an event based attention that takes into account the hierarchical structure of sentences.

7 Conclusion

We present serveral techniques to study the explainability of sequential classification models. Model explanations are evaluated on three datasets, where it is observed that the attention-based methods outperformed the gradient-based methods. To understand the differences in explanations between trained models initialized with different initial seeds, we develop a low-loss trail algorithm to determine a low-loss path between trained models in order to understand how explanations change between models. Finally, by examining explanations along the low-loss path, we can classify high attention event types and provide an explanation for differences in attention clustering between trained models initialized with different seeds.

References

[1] Marco Ancona, Enea Ceolini, Cengiz Oztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *6th International Conference on Learning Representations*, 2018.

[2] Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. "what is relevant in a text document?": An interpretable machine learning approach. *PloS One*, 12(8):e0181142, 2017.

[3] Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. Explaining recurrent neural network predictions in sentiment analysis. *arXiv preprint arXiv:1706.07206*, 2017.

[4] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS One*, 10(7):e0130140, 2015.

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

[6] Inci M Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K Jain, and Jiayu Zhou. Patient subtyping via time-aware LSTM networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 65–74, 2017.

[7] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[8] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pages 3504–3512, 2016.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] Yannis Dimopoulos, Paul Bourret, and Sovan Lek. Use of some sensitivity criteria for choosing networks with good generalization ability. *Neural Processing Letters*, 2(6):1–4, 1995.

[11] Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, 2017.

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

[13] Andrej Karpathy. The unreasonable effectiveness of recurrent neural networks. http://karpathy.github.io/2015/05/21/rnn-effectiveness/, 2015.

[14] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*, 2015.

[15] Rohith Kuditipudi, Xiang Wang, Holden Lee, Yi Zhang, Zhiyuan Li, Wei Hu, Rong Ge, and Sanjeev Arora. Explaining landscape connectivity of low-cost solutions for multilayer nets. *Advances in Neural Information Processing Systems*, 32, 2019.

[16] Harold W Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[17] Edward Loper and Steven Bird. NLTK: the natural language toolkit. *arXiv preprint cs/0205028*, 2002.

[18] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.

[19] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 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, 2011.

[20] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52, 2015.

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[22] James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. Explainable prediction of medical codes from clinical text. *arXiv preprint arXiv:1802.05695*, 2018.

[23] W James Murdoch, Peter J Liu, and Bin Yu. Beyond word importance: Contextual decomposition to extract interactions from LSTMs. *arXiv preprint arXiv:1801.05453*, 2018.

[24] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016.

[25] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[26] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*, 2017.

[27] Jorge M Santos and Mark Embrechts. On the use of the adjusted rand index as a metric for evaluating supervised classification. In *International Conference on Artificial Neural Networks*, pages 175–184, 2009.

[28] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017.

[29] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

[30] G Tian, L Tao, and L Yao. An interpretable lstm neural network for autoregressive exogenous model. In *ICLR 2018 Workshop*, 2018.

[31] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.

A Low-Loss Trail Algorithm

The algorithm is exhibited next.

Algorithm 1 Low-Loss Trail Algorithm

Input: weights w^1, w^2, $L_{\max}$, ε, J
Parameters: $\alpha_{\min} \geq 0$, $0 < \tau < 1$, $\bar{\delta} > 0$, $\bar{\varepsilon} > 0$, $\rho > 1$
Output: a sequence of weights $\hat{w}$
Initialization: $\hat{w}_1 = w^1$

1: **for** $k = 1, 2, 3, \ldots$ **do**
2: $\quad w = (1-J)\hat{w}_k + Jw^2$
3: $\quad$ **if** $\ell(w) \leq L_{\max} + \varepsilon$ **then**
4: $\quad\quad \hat{w}_{k+1} = w$
5: $\quad$ **else**
6: $\quad\quad$ Solve $Dir(w)$ by Lagrangian and let d be the optimal solution
7: $\quad\quad \alpha = \frac{J||w^2 - w||}{||d||}$
8: $\quad\quad \bar{w} = w - \alpha d$
9: $\quad\quad$ **while** $\ell(\bar{w}) \leq L_{\max} + \varepsilon$ and $\alpha \leq \alpha_{\min}$ **do**
10: $\quad\quad\quad \alpha = \tau \cdot \alpha$
11: $\quad\quad\quad \bar{w} = w - \alpha d$
12: $\quad\quad$ **end while**
13: $\quad\quad \hat{w}_{k+1} = \bar{w}$
14: $\quad\quad$ **if** $\ell(\hat{w}_{k+1}) > L_{\max} + \varepsilon$ or $||\hat{w}_k - w^2|| - ||\hat{w}_{k+1} - w^2|| \leq \bar{\delta}$ **then**
15: $\quad\quad\quad \varepsilon = \rho\varepsilon$
16: $\quad\quad$ **end if**
17: $\quad$ **end if**
18: $\quad$ **if** $||\hat{w}_{k+1} - w^2|| \leq \bar{\varepsilon}$ **then**
19: $\quad\quad$ break
20: $\quad$ **end if**
21: **end for**

Parameter $\alpha_{\min}$ is the smallest step size during line search while $\tau < 1$ is the step size reduction. Value $\rho > 1$ governs the increase of ε when no progress is made. Finally, $\bar{\varepsilon}$ is the stopping criterion and $\bar{\delta}$ controls when the increase of ε is triggered.

It is easy to see that the algorithm always meets (2) but since in Step 15 ε is increased, it might satisfy (1) with higher tolerance.

Vector w is the vector of next weights. In steps 2-4 we follow the direct line to w^2 while the loss is below the target. As soon as it is too high, in Step 6 we compute a descending direction d. In Steps 9-11 we perform line search. In Step 14, if line search yields "non-acceptable" loss or not enough progress is made towards reaching w^2, we increase the allowable loss value.

B Explainability

B.1 Amazon Polarity

Run	**0**	**1**	**2**
0	1	0.25	0.31
1	0.25	1	0.20
2	0.31	0.20	1

(a) ARI Scores for standard attention explainability

Run	**0**	**1**	**2**
0	1	0.34	0.40
1	0.34	1	0.38
2	0.40	0.38	1

(b) ARI Scores for hierarchical attention explainability

Table 18: Clustering Results.

Here we use two clusters and they are derived from silhouette plots.

Run	**0**	**1**	**2**
0	1	0.500	-0.001
1	0.500	1	-0.001
2	-0.001	-0.001	1

(a) ARI Scores for gradient explainability

Run	**0**	**1**	**2**
0	1	1.00000	-0.00062
1	1.00000	1	-0.00062
2	-0.00062	-0.00062	1

(b) ARI Scores for integrated gradient explainability

Table 19: Clustering Results.

B.2 IMDB

Run	**0**	**1**	**2**
0	1	0.21	0.21
1	0.21	1	0.24
2	0.21	0.24	1

(a) ARI Scores for attention explainability

Run	**0**	**1**	**2**
0	1	0.14	0.19
1	0.14	1	0.20
2	0.19	0.20	1

(b) ARI Scores for hierarchical attention explainability

Table 20: Clustering Results.

Run	0	1	2
0	1	-0.00043	0.29000
1	-0.00043	1	-0.00049
2	0.29000	-0.00049	1

(a) ARI Scores for gradient explainability

Run	0	1	2
0	1	-0.00064	-0.00056
1	-0.00064	1	-0.00078
2	-0.00056	-0.00078	1

(b) ARI Scores for integrated gradient explainability

Table 21: Clustering Results.

B.3 Business Use Case

Run	0	1	2
0	1	0.04	0.08
1	0.04	1	0.48
2	0.08	0.48	1

(a) ARI Scores for attention explainability

Run	0	1	2
0	1	0.20	0.20
1	0.20	1	0.11
2	0.20	0.11	1

(b) ARI Scores for hierarchical attention explainability

Table 22: Clustering Results.

We note that there is a large difference in ARI scores for the standard attention models.

Run	0	1	2
0	1	0.000056	0.330000
1	0.000056	1	0.000092
2	0.330000	0.000092	1

(a) ARI Scores for gradient explainability

Run	0	1	2
0	1	0.00024	0.00045
1	0.00024	1	0.58000
2	0.00045	0.58000	1

(b) ARI Scores for integrated gradient explainability

Table 23: Clustering Results.

As we observe in the other datasets, the gradient explainability mechanism returns low ARI scores for most pairs of models. When we examine the clusters for the two model runs that return an ARI score of 0.33 in Table 23a, we observe that for one of the model runs, over 95% of the samples belong to the same class. This suggests that gradient-based methods are unsuitable for event type based explanations. Similarly, we

consider the clusters for the two model runs that return an ARI score of 0.58 in Table 23b, we note that 96% of the samples belong to the same class for one of the models. This suggests that integrated gradient-based methods suffer from the same problems as the gradient-based methods.