# Combining Unsupervised Pre-training and Annotator Rationales to Improve Low-shot Text Classification

**Oren Melamud**
IBM Research
Yorktown Heights, NY, USA
oren.melamud@ibm.com

**Mihaela Bornea**
IBM Research
Yorktown Heights, NY, USA
mabornea@us.ibm.com

**Ken Barker**
IBM Research
Yorktown Heights, NY, USA
kjbarker@us.ibm.com

**Abstract**

Supervised learning models often perform poorly at *low-shot* tasks, i.e. tasks for which little labeled data is available for training. One prominent approach for improving low-shot learning is to use unsupervised pre-trained neural models. Another approach is to obtain richer supervision by collecting annotator *rationales* (explanations supporting label annotations). In this work, we combine these two approaches to improve low-shot text classification with two novel methods: a simple bag-of-words embedding approach; and a more complex context-aware method, based on the BERT model. In experiments with two English text classification datasets, we demonstrate substantial performance gains from combining pre-training with rationales. Furthermore, our investigation of a range of train-set sizes reveals that the simple bag-of-words approach is the clear top performer when there are only a few dozen training instances or less, while more complex models, such as BERT or CNN, require more training data to shine.

## 1 Introduction

When trained on large amounts of labeled data, supervised machine learning models demonstrate impressive performance in various domains ranging from text classification to image recognition (Yogatama et al., 2017; Krizhevsky et al., 2012). In many domains, however, labeled data is difficult to obtain. This might be due to annotation cost, or simply because there are no available instances to annotate before the system has to make the next classification decision. Imagine, for example, a personalized email tagger, where new tag types can be created and then used for annotation by the user at any time. When only little training is available, model performance drops dramatically.

Various methods have been proposed to optimize machine learning models trained on little data with the goal of improving performance for a given train set size or reducing the number of training instances that need to be collected. One of the most prominent for text classification (and other NLP tasks) is transfer learning from unsupervised data, or unsupervised pre-training. This approach has been widely adopted with the introduction of pre-trained word embeddings (Mikolov et al., 2013; Joulin et al., 2017). Word type representations have subsequently been extended with pre-trained, context-dependent representations achieving strong results across a wide range of NLP tasks (Melamud et al., 2016; Peters et al., 2018; Howard and Ruder, 2018). A recent strong performer in this line of research is BERT (Devlin et al., 2018), a multi-layer attention-based neural model that is pre-trained on large amounts of plain text and then fine tuned to specific tasks. Systems using BERT have achieved state-of-the-art performance on various NLP tasks, including short text classification.

Another approach for training text classifiers with few labeled instances is to supplement the class labels with annotation rationales. Annotation rationales for text are usually specific subsequences of words within a text instance that the annotator considers to be evidence supporting the label assignment. Zaidan et al. (2007) were the first to investigate the use of annotator rationales for training text classifiers. They obtained manual rationales for an IMDB movie review dataset (Pang et al., 2002) and showed that an SVM classifier can be modified to obtain improved accuracy using these rationales. The bold text in the movie review snippet "*perhaps* ***the greatest element to the film*** *is its constant surprises and unpredictability*" is one example of an annotated rationale in this dataset. Abedin et al. (2011) showed similar results using SVM and rationales for classifying cause identifiers in aviation safety reports from the Aviation Safety Reporting System (ASRS).

In this work, we combine the power of unsupervised pre-training with the fine-grained guidance from rationales to propose two novel low-shot text classification methods. Both methods promote rationale-like features in the input to improve the classification outcome. The first method uses rationale supervision to bias a simple bag-of-words text embedding towards the more discriminative features. In the second method, we show how to fine-tune a pre-trained BERT model with both instance labels and rationales jointly. An investigation across a range of train-set sizes on the IMDB and ASRS text classification datasets reveals that our bag-of-words approach substantially outperforms all baselines for train sets of up to a few dozen instances. For larger sizes, our BERT-based model and a CNN baseline perform the best. Our code is publicly available. [1]

## 2 Related Work

As mentioned in the Introduction, Zaidan et al. (2007) was the first work to show how manual rationales annotated for movie reviews (Pang et al., 2002) can be used to improve classifier accuracy. They also investigated the annotation effort ramifications and found that the additional annotation of a few rationales per instance on top of the standard label annotation roughly doubled annotation time. When enough instances are readily available for annotation, this information is important to assess the trade-off between obtaining annotator rationales and obtaining standard labeled instances. However, we note that there are cases where more unlabeled instances may not be available and then obtaining annotator rationales may be the only way to improve the classifier accuracy.

The model most closely related to our rationale-biased bag-of-words is the one proposed by Sharma and Bilgic (2018) (denoted RA-SVM in Section 5.2). It biases the input representation by adjusting feature weights in a way that is agnostic to the type of classifier used. More specifically, it uses one-hot word features and performs simple fixed discounting of non-rationale word feature weights during training only. In contrast, our model learns a rationale-biased embedded representation of texts and this representation is used both at train and test time.

The model most closely related to our rationale-biased BERT model is the one proposed by Zhang et al. (2016) (denoted RA-CNN in Section 5.2). This model uses a CNN sentence encoder with pre-trained word embeddings. It follows a two-step approach, where one sentence classifier is trained to identify sentences containing rationales and weigh them higher in an overall 'weighted-average-of-sentences' document representation. Then a second classifier is trained to make the final label prediction based on that representation. They showed improved accuracy compared to non-rationale-augmented CNNs as well as to rationale-augmented SVM models. Our main contributions relative to that work are the introduction of the joint-learning technique, which learns text classification and rationale words identification concurrently, and the adaptation of the entire approach to BERT. The ability of our model to identify rationale spans within sentences also seems useful for interpretability. Tepper et al. (2013) also chose a two-step approach similar to Zhang et al. (2016) using more traditional classifiers with sparse features. They train one classifier to identify rationale-like pieces of text and then a second classifier is trained to label input instances taking only the rationale-like texts into account.

Finally, *few-shot* learning methods attempt to learn how to effectively train classifiers for new classes with only few labeled instances, using techniques commonly referred to as *meta-learning* (Snell et al., 2017; Yu et al., 2018). Snell et al. (2017), for example, learn an embedding function that is used to represent the input instances such that simple nearest-prototype classifiers can be trained effectively to recognise instances of a new target class after being exposed to a small number of labeled examples. Although a new class is learned from few labeled instances as in our work, the preceding meta-learning process relies on a large number of labeled instances aggregated over many closely related classes or tasks. This is an important distinction from our low-shot setting.

## 3 Rationale-biased Bag-of-words

In this section, we describe our proposed rationale-biased bag-of-words method for performing low-shot text classification with rationales.

[1] https://github.com/mihaela-bornea/low-shot-text-classification

### 3.1 Rationale-biased representation

Let $\vec{v}$ be a vector representation of word $v$ (either one-hot or a low-dimensional embedding). We represent a text instance $t = (v_1, .., v_n)$ as an L2-normalized, weighted average of its words giving the text embedding function:

$$emb(t) = \vec{t} = \frac{1}{Z} \sum_{v_i \in t} w_i \vec{v_i} \qquad (1)$$

where $Z$ is the normalization factor.

This embedding is a standard bag-of-words approach to representing texts, where weights, such as TF-IDF, are typically used to bias the representation towards words that carry more important information. We hypothesize that since rationales highlight support for annotator classification decisions, biasing this text embedding function towards *rationale-like* words would expose the more discriminative features. To this end, given a rationale-bias function $f_{rb}(v)$ that estimates the similarity between a word $v$ and the annotated rationale texts in the training dataset, the rationale biased representation could be computed using Eq. (1) with $w_i = f_{rb}(v_i)$. The remainder of this section describes how we compute $f_{rb}(v)$.

Let the *prototype* representation of a collection of text instances $T = (t_1, .., t_m)$ be the L2-normalized centroid of its text instance vectors:

$$prot(T) = \frac{1}{Z} \sum_{t_i \in T} \vec{t_i} \qquad (2)$$

where $Z$ is the normalization factor.

A rationale for a text instance $t$ and class label $c$ is any word subsequence $r$ of $t$ that was annotated as evidence for the assignment of $c$ to $t$. A *class rationale prototype* $\vec{R_i} = prot(R_i)$ is the prototype representation of the rationales $R_i = (r_1, .., r_l)$ annotated for class $c_i$, where each rationale instance $r$ is embedded using Eq. (1) with uniform weights. We define the following rationale-bias function $f_{rb}(v)$:

$$f_{rb}(v) = (\exp sim_{rb}(v))^{\alpha} \qquad (3)$$

$$sim_{rb}(v) = \max_{i \in \{1..k\}} cosine(\vec{v}, \vec{R_i})$$

That is, $sim_{rb}(v)$ is the maximum cosine similarity of the word vector to any of the class rationale prototypes, and $f_{rb}(v)$ is an always positive derivative of $sim_{rb}(v)$ that is controlled by the hyperparameter $\alpha \geq 0$. Higher values of $\alpha$ mean stronger magnification of the impact of the rationale biases; $\alpha = 0$ means that the rationale bias is completely ignored. With this rationale-bias function, we can now compute rationale-biased representations for text instances using Eq. (1), and these representations can be used as inputs to any classifier.

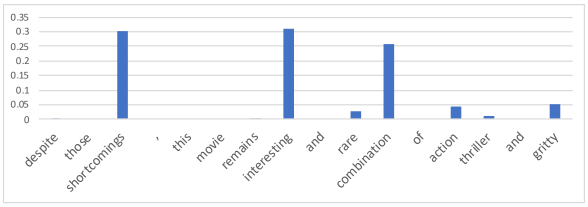

Figure 1: The bias weights generated by a rationale-biased model trained on 20 instances, to a snippet from an example movie review.

In early experiments, we noticed that many *non-discriminating*, common words in the target domain (e.g., *movie*, *certainly*, *guy*) tended to get high rationale biases. This is presumably due to the fact that rationales in these datasets often have spans that include multiple rationale words along with other words. To compensate, we adjusted the class rationale prototype representation to:

$$adjusted(R_i) = prot(R_i) - prot(T_{all}) \qquad (4)$$

where $T_{all}$ is the collection of all training instances across all classes. That is, we subtract the entire training-set prototype vector from each class rationale prototype. We L2-normalize this representation as well.

Figure 1 is an example illustrating actual bias weights generated by our model. As can be seen in this case, the most positive and negative indicative words are 'picked up' as indicated by the high respective bias. However, it should also be noted that this model misses the sentiment polarity inversion expressed by the word *despite*, due to its simple sequence-agnostic nature.

### 3.2 Classifiers

Various types of classifiers can be trained to perform text classification on top of the rationale-biased text representations described in the previous section. We chose to experiment with a standard SVM classifier as well as a simple, nearest-prototype classifier similar to the approach taken in prior few-shot learning work (Snell et al., 2017). Before training and applying these classifiers, we compute the class rationale prototypes to determine the rationale-bias function $f_{rb}$. We then use that function to embed all of the dataset text instances and use these text embeddings as inputs to the classifiers. More specifically, for our nearest-prototype classifier, given a training set of text instance collections $T_1, .., T_k$ labeled with classes $c_1, .., c_k$, we represent each class $c_i$ by the *class prototype* $\vec{T_i} = prot(T_i)$. To classify a new text instance $t'$ we simply choose the closest class prototype using cosine similarity:

$$class(t') = \arg\max_{i \in \{1..k\}} cosine(\vec{t'}, \vec{T_i}) \tag{5}$$

Our hypothesis is that this proposed method would do comparatively well with very little training data because (a) with the guidance of the rationale bias it could pick up individual discriminative words more quickly (e.g., *good* or *bad* for sentiment tasks); (b) combining the use of pre-trained word embeddings as the underlying word representation with rationale bias would help generalize to words unseen in the training data (e.g. to *best* from *good*) and would facilitate an effective rationale bias similarity function $sim_{rb}(v)$; and (c) the training regimen is simple with a relatively small number of parameters to be learned compared to some more complex neural models such as the one discussed in the following section.

## 4 Rationale-biased BERT

BERT (Devlin et al., 2018) is a context-aware neural network model that has shown excellent results when used as a basis for various supervised NLP systems. Its power comes from transferring context-aware language modeling information learned from unsupervised training over very large amounts of plain text. In this section, we show how we adapt BERT to handle long texts and then to incorporate rationale supervision together with standard instance labels.

### 4.1 Fine-tuning BERT

Presented with a sequence of word tokens[2] $s = (a_1, ..a_n)$ as input, a BERT model outputs contextualized vector encodings for each token $B(s) = (h_0, h_1, .., h_n)$,[3] where $h_0$ is a special encoding for the entire sequence. Pre-trained versions of this model were trained by Devlin et al. (2018) against language modeling and sentence prediction objectives on large corpora of text allowing them to capture valuable contextualized information in these encodings. Devlin et al. (2018) applied dropout followed by a simple linear layer, $L_{tok}(drop(B(s)_i))$ where $i > 0$ to perform token-level classification and $L_{text}(drop(B(s)_0))$ for sentence-level classification. They fine-tuned the entire model on task-specific labels using a standard cross-entropy loss function. With this approach, they achieved state-of-the-art results on various NLP tasks, such as Named Entity Recognition (token-level) and single-sentence sentiment classification (sentence-level).

[2]BERT breaks words into subword units called *word pieces* (Wu et al., 2016).

[3]We refer here to the top-most output layer of BERT.

### 4.2 Averaged BERT classifier for long texts

While BERT proved to be very effective for classifying short texts such as individual sentences, its computational complexity is quadratic with the length of the input text yielding long run times for texts longer than a few sentences. Since our input texts may be long, we split them into sentences, apply BERT to each of the sentences separately and average the outputs as follows.

Given a text instance comprising the set of sentences $t = (s_1, ..., s_m)$, we encode it as the weighted average of the individual sentences:

$$B_{avg}(t) = \vec{t} = \sum_{i \in 1..m} w_i \cdot drop(B(s_i)_0) \tag{6}$$

where $w_i$ are the weights used in the weighted average. We experiment with uniform weights and also with learning weights as learned attention, using a linear layer $L_{attn}$:

$$w_i = softmax(L_{attn}(drop(B(s_i)_0))) \tag{7}$$

Finally, similar to the original BERT approach, to fine tune a classifier for long text instances, we apply a linear layer to the text encoding $L_{text}(B_{avg}(t))$ and train the whole model to predict the target labels with the loss function:

$$S_{target}(t, l_t) = CE(L_{text}(B_{avg}(t)), l_t) \tag{8}$$

where $l_t$ is the training label for text instance $t$ and $CE$ is the cross-entropy loss function. Figure 2 illustrates our architecture.

### 4.3 Rationale-biased BERT classifiers

We propose two methods to incorporate rationale supervision into our averaged BERT text classifier. The first is based on straightforward multi-task learning, where rationale word prediction is considered an auxiliary task that is jointly trained

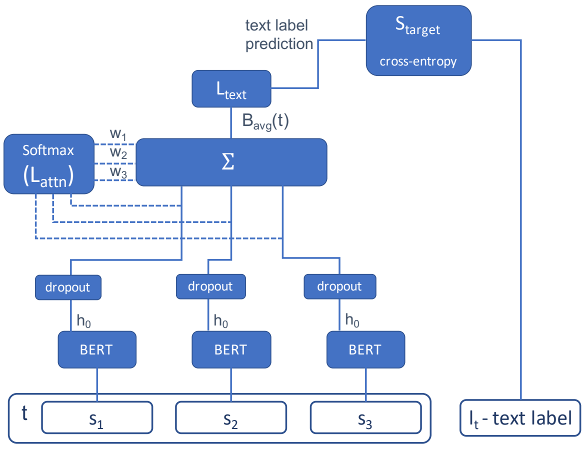

Figure 2: Averaged BERT model architecture. The three BERT boxes refer to the same single instance of the model. $L_{attn}$ is the optional sentence attention learning component from Eq. (7).

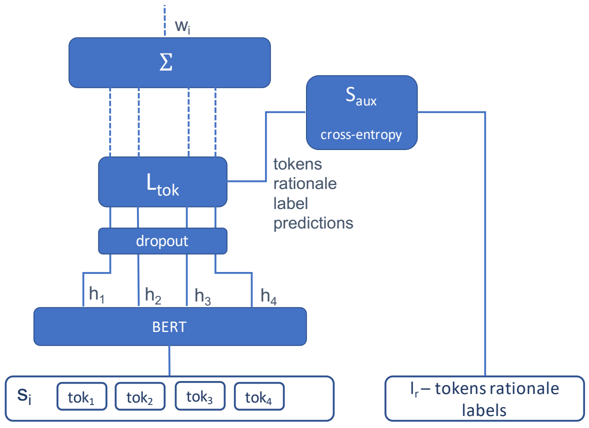

Figure 3: Rationale-bias auxiliary model component. The BERT model here is the same one used for the main target task. $w_i$ is the average probability that a token in the sentence is part of a rationale, which can be used as sentence attention in the averaged BERT.

with the main text label prediction task. The second extends the first by using rationale supervision to estimate the attention to each sentence. It then uses that attention (or importance measure) to inform the weighted average from Eq. (6).

**Simple joint learning.** Figure 3 illustrates the auxiliary model added to the averaged BERT. The same BERT model is trained to perform two classification tasks: (1) predicting the label of each text instance (target text classification task); and (2) predicting for every input token in every input sentence whether it is part of an annotated rationale subsequence (auxiliary task). This is achieved by introducing the auxiliary loss function:

$$S_{aux}(t, l_r) = \frac{1}{n}\sum_{i,j} CE(L_{tok}(drop(B(s_i)_j)), l_r^{i,j}) \tag{9}$$

where $l_r^{i,j}$ is the rationale label of token $j$ in sentence $i$, $L_{tok}$ is a linear layer for learning to classify tokens and $n$ is the total number of tokens in text $t$. Finally, the multi-task objective function per input text $t$ is:

$$S_{joint}(t, l_t, l_r) = S_{target}(t, l_t) + S_{aux}(t, l_r) \tag{10}$$

Our hypothesis is that the auxiliary task will infuse lower-level information regarding the relevance of input words into the BERT model, which will help boost the main task classifier.[4]

[4]In early experiments, we also tried infusing this information to lower layers of the BERT model (i.e., to an internal layer rather than the output layer) following Søgaard and Goldberg (2016), but that did not improve the performance.

**Rationale-biased Attention.** The dotted lines in Figure 3 illustrate our proposed approach to extend simple joint-learning by using the rationale signal to learn how to identify the more important sentences in the input text. To train this model, we use the same multi-task loss function in Eq. (10). However, this time, instead of using uniform weights in the averaged BERT, we consider the attention weight of a sentence to be the average of the probabilities that each of its tokens is in a rationale span. These weights are used to compute the final weighted average representation of the text input in Eq. (6). In our experiments, we found that it was better not to backpropagate the error from $S_{target}$ towards the token-level rationale classifier (i.e. along the dotted lines). Therefore, the rationale classifier is still trained only based on $S_{aux}$.

The hypothesis here is that this model could be successful by infusing more explicitly the inductive bias that sentences with rationale words are more important than others for the final classification task. Incorporating more inductive bias is particularly important with little training data. Compared to the bag-of-words approach presented in the previous section, however, we presume that both BERT-based methods presented here might be more challenged with extremely little training data because of the more complex underlying models and training regimen.

Figure 4 shows an example of sentence attentions and word rationale probabilities computed by our rationale-biased attention BERT model.

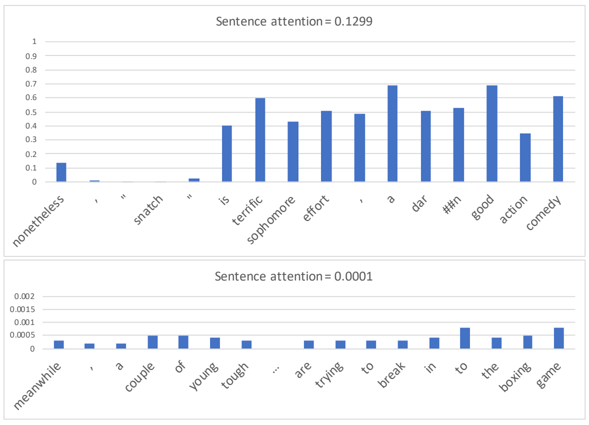

Figure 4: Excerpts from the sentence that got the highest attention (top) and lowest attention (bottom) in an IMDB movie review, by our rationale-biased attention BERT model. Bars indicate the probability the model assigns for every word of being part of a rationale.

Being context-aware, this model captures relevant sequences of words rather than just spotting key words. In addition to being useful for the classification task, these sentence and token weights also seem useful for model interpretability.

## 5 Experimental Settings

We conducted experiments with two English text classification datasets. Zaidan et al. (2007) added rationales supporting the positive/negative class labels on 1800 movie reviews from the Internet Movie Database (IMDB) dataset (Pang et al. (2002)). We used 900 reviews for training, 450 for development and 450 for testing, and maintained the original even positive/negative balance in every split.

The Aviation Safety Reporting System (ASRS) dataset contains reports of aviation safety incidents. We used the version from Abedin et al. (2011), which has 1233 reports in the training set, 100 for development and 1000 for testing. Reports are labeled with one or more of 14 safety incident cause categories. To create a binary classification dataset comparable to the IMDB dataset, we extracted a balanced subset of reports that were labeled with the 'Proficiency' label or the 'Physical Environment' label (but not both). We chose these particular two labels because they have a similar number of instances and are among the top most frequent labels in the dataset. This resulted in 386 reports for training and 392 reports for testing. Table 1 provides some additional statistics regarding these two datasets.

| | IMDB | ASRS |
|---|---|---|
| sentences per instance | 33.8 | 15.3 |
| words per instance | 736.6 | 262.1 |
| rationales per instance | 8.6 | 3.0 |
| words per rationale | 7.2 | 2.2 |

Table 1: Dataset statistics

### 5.1 Experimental protocol

To test the hypothesis that our proposed rationale-biased methods can improve classification with little training data, we ran experiments varying the amount of training data used from 2 instances up to roughly 400. For each training size $n$, we sampled $n$ training instances to train the models and tested on the entire dev/test set. We repeated this experiment 30 times (for each $n$) and report average accuracies. We fixed all parameters based on tuning on only the IMDB development set prior to running on the IMDB and ASRS test data (no tuning was done on ASRS data).

### 5.2 Compared methods

***RA-SVM*** is the Rationale-Augmented SVM by Sharma and Bilgic (2018) that biases the text representation to rationales by discounting non-rationale word features by a fixed discount ratio during training. All words are TF-IDF weighted prior to discounting. We experimented with different discounts on the IMDB development set and found that 0.1 gave the best results. And while Sharma and Bilgic (2018) use a one-hot word representation, we also experimented with giving it dense word embeddings, but this yielded degraded performance. We also report results with no rationale discount - a plain ***SVM*** baseline.

***RA-CNN*** is the Rationale-Augmented CNN by Zhang et al. (2016), described in section 2. We used the authors' implementation with pre-trained word2vec Google News word embeddings with 300 dimensions for 3 million words.[5] We tuned batch size, dropout, learning rate and number of epochs on our development set. We also report results with no rationale weighting - a plain ***CNN*** baseline.

***RB-BOW-SVM*** and ***RB-BOW-PROTO*** are the SVM and nearest-prototype classifiers running on top of our proposed rationale-biased bag-of-words (BOW) text representation (Section 3). We tuned

[5]https://code.google.com/archive/p/word2vec/

the hyperparameter $\alpha$ on the dev set, obtaining the best results with $\alpha = 6$ for both classifiers. ***BOW-PROTO*** is the prototype classifier based on a BOW with no rationale bias ($\alpha = 0$). We tried both one-hot word representations and word embeddings using the same word2vec embeddings as for CNN and RA-CNN.

***AVG-BERT*** and ***ATTN-WAVG-BERT*** are our proposed BERT-based models trained without rationales using a uniform-weight average and an attention-based weighted average of sentences, respectively (Section 4.2). ***RB-AVG-BERT*** is the joint model that uses rationale prediction as an auxiliary task, and ***RB-WAVG-BERT*** is the method that also uses rationales to estimate sentence-level attention (Section 4.3). For all of the above, we used the pre-trained models 'bert-base-uncased' and 'bert-base-cased' for the IMDB and ASRS datasets, respectively. We used default hyperparameters, except for using learning rate of 5e-6 and 10 epochs for fine-tuning, which we found to work better for AVG-BERT with light tuning. To fit the computation graphs into the GPU memory (NVIDIA Tesla V100) we trimmed sentences longer than 48 word pieces, and texts containing more than 64 sentences. We implemented our models using a PyTorch implementation of BERT.[6]

***ULMFiT*** (Howard and Ruder, 2018) is a recent method achieving state-of-the-art text classification results on several datasets. It is trained in three steps: (1) training a general-domain recurrent neural network language model on a large corpus (WikiText-103); (2) fine-tuning the language model to the domain data of the target task disregarding class labels; and (3) fine-tuning a classifier for the task using the encoder of the learned fine-tuned language model as a starting point. We used the *fast.ai*[7] implementation of ULMFiT, tuning batch size, dropout multiplication factor, learning rates and number of epochs on our development set. ULMFiT does not currently have a rationale-augmented version.

## 6 Results

In this section, we start by investigating the performance of different variants of our proposed models on the IMDB dev set. The most promising configurations are then evaluated on the IMDB and ASRS test sets. Since both of our datasets are binary and balanced we note that the random baseline accuracy is 50%.

### 6.1 Development set investigation

To demonstrate the contribution of various components of our rationale-biased bag-of-words approach, we performed several ablation experiments on the development set, shown in Figure 5. The compared variants include: ***RB-BOW-PROTO***: our full method with pre-trained word embeddings; ***BOW-PROTO***: no rationale-bias; ***OH***: RB-BOW-PROTO with one-hot word representations instead of word embeddings; ***NO-ADJUST***: RB-BOW-PROTO without discounting common words by subtracting the general prototype vector (Eq. (4)); ***SINGLE***: RB-BOW-PROTO with just a single rationale prototype representing all rationales instead of one rationale prototype for each class (Eq. (3)); and ***INSTANCE***: RB-BOW-PROTO with a rationale prototype for every individual text instance instead of one per class. As can be seen, combining the use of pre-training and rationales is critical to the success of our full method. First, the importance of using pre-trained word embeddings is evident by the significant drop in performance when using one-hot encodings with train-sets with 60 instances or fewer. Second, the importance of rationale-biasing can be seen by the large performance drop with BOW-PROTO across all train-set sizes. With those two components in place, we see that the RB-BOW-PROTO is able to perform significantly above the 50% random baseline with very few training instances. We note that RB-BOW-SVM demonstrated similar performance to RB-BOW-PROTO. For brevity, we don't include a detailed report here, but do include RB-BOW-SVM in the test results section.

To illustrate the potential merit in using our proposed rationale-biased embedding function we show 2-dimensional plots of text embeddings with and without rationale-bias in Figure 7. We randomly sampled a train-set of size 60 from the IMDB dataset and used it to learn the rationale-biased embedding function. We plotted the positive and negative class prototypes as learned from the training set as well as the embeddings of the IMDB dev set instances. We used T-SNE to reduce those 300-dimensional embeddings down to

[6]https://github.com/huggingface/pytorch-pretrained-BERT
[7]https://www.fast.ai/

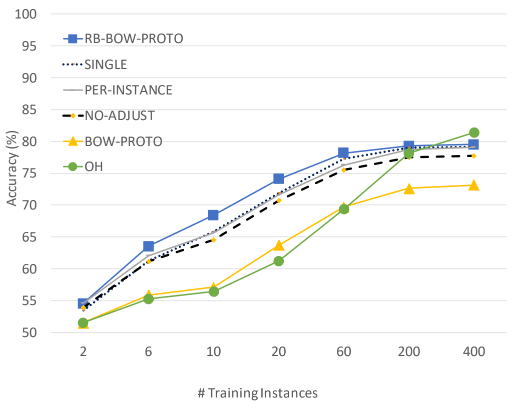

Figure 5: Ablation experiments on the IMDB dev set.

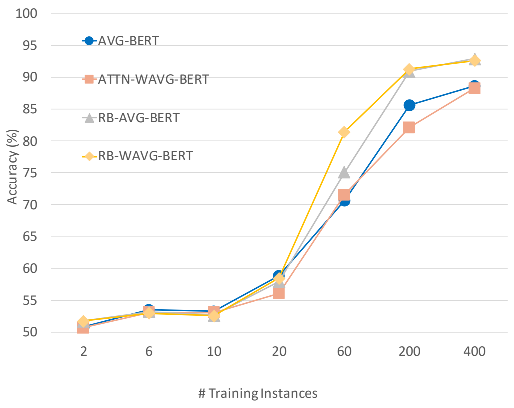

Figure 6: BERT-based models on the IMDB dev set.

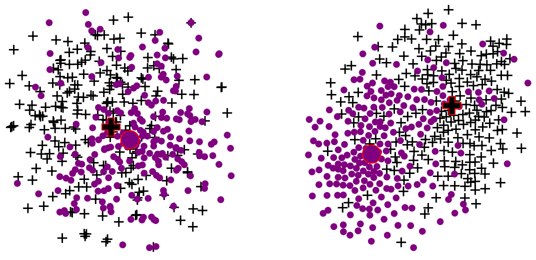

Figure 7: Bag-of-words text embedding without rationale bias (left) and with (right). +/o markers are positive/negative dev text instances. Large markers are positive/negative class prototypes learned during training.

two dimensions for visualization. The rationale-biased representation appears to discriminate better between positive and negative instances. In particular, we notice an increase in the distance between the representations of class prototypes and a clearer clustering of the development set instances around their respective class prototypes. For this data subset, the nearest prototype classifier achieves accuracy of 78.1% when using the rationale-biased embedding compared to 69.7% with an unbiased representation.

Next, we investigate the performance of the BERT-based models on the IMDB dataset. Figure 6 shows the results on the dev set. First we note that unlike the simpler bag-of-words methods, all of these models require at least 20 training instances before they meaningfully improve on the random 50% baseline. On the other hand, with 200 training instances or more, all of the BERT-based models significantly outperform the bag-of-words methods. More specifically, we see that RB-AVG-BERT and RB-WAVG-BERT, which use rationale supervision, significantly outperform AVG-BERT and ATTN-WAVG-BERT, which only use text label supervision, by up to 10 accuracy points. Between RB-AVG-BERT and RB-WAVG-BERT, the latter, which incorporates more inductive bias, performs more robustly.

### 6.2 Test set results

Figure 8 and Table 2 show the IMDB test-set results of our best rationale-biased methods and their non-rationale counterparts (chosen based on dev set performance), as well as all of the baselines.[8] Our bag-of-words models, RB-BOW-PROTO and RB-BOW-SVM, which combine rationale-biasing with pre-trained word embeddings, outperform all other baselines by a large margin of up to more than 20 absolute accuracy points for training sets of size 20 or smaller. On these particularly small datasets, the other rationale-aware baselines, RA-SVM and RA-CNN, do not perform as well. For larger training sets of size 60 and more, all systems perform better, as expected. In particular, the more complex CNN and BERT-based models have significantly improved performance with more supervision. Both RA-CNN and our RB-BERT, which combine pre-training with rationales, benefit substantially from rationale supervision, with up to more than 30 absolute accuracy points improvement. In absolute terms, our BERT models outperform their CNN counterparts and every other baseline in this training size range.

Finally, Figure 9 and Table 3 show results on the ASRS test set. The trends here are similar, showing the robust merit in our rationale-bias approach. The only striking difference is that in this dataset, the CNN methods outperform our BERT-based

[8]ULMFiT is not included in Figures 8 and 9 for better readability.

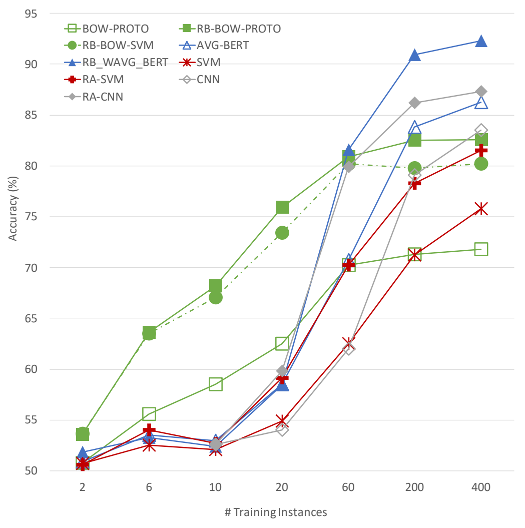

Figure 8: IMDB test set results

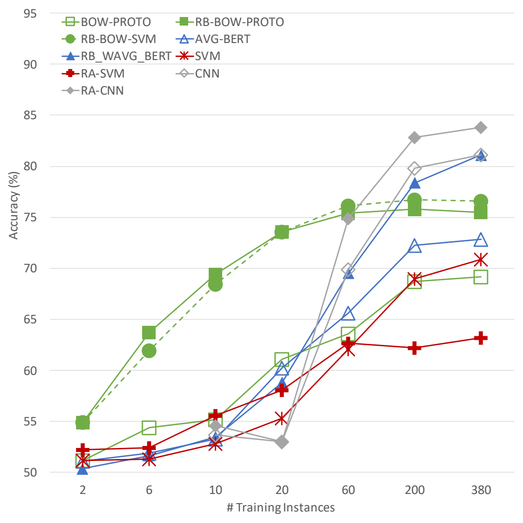

Figure 9: ASRS test set results

| | Number of Training Instances | | | | | | |
|---|---|---|---|---|---|---|---|
| | **2** | **6** | **10** | **20** | **60** | **200** | **400** |
| **RB-BOW-PROTO** | 53.6 | 63.6 | 68.2 | 75.9 | 80.9 | 82.5 | 82.6 |
| **RB-BOW-SVM** | 53.6 | 63.5 | 67.0 | 73.4 | 80.2 | 79.7 | 80.2 |
| **BOW-PROTO** | 50.8 | 55.6 | 58.5 | 62.5 | 70.2 | 71.3 | 71.8 |
| **RB-WAVG-BERT** | 51.8 | 53.2 | 52.3 | 58.4 | 81.6 | 90.9 | 92.2 |
| **AVG-BERT** | 50.9 | 53.5 | 52.9 | 58.5 | 70.7 | 83.8 | 86.2 |
| **RA-SVM** | 50.6 | 54 | 52.7 | 59.1 | 70.2 | 78.3 | 81.5 |
| **SVM** | 50.7 | 52.5 | 52.1 | 54.9 | 62.5 | 71.2 | 75.8 |
| **RA-CNN** | * | * | 52.6 | 59.8 | 79.9 | 86.2 | 87.3 |
| **CNN** | * | * | 52.6 | 54 | 62 | 79.1 | 83.5 |
| **ULMFiT** | * | * | * | 54.7 | 62 | 71.3 | 78.8 |

Table 2: IMDB test set results.

| | Number of Training Instances | | | | | | |
|---|---|---|---|---|---|---|---|
| | **2** | **6** | **10** | **20** | **60** | **200** | **380** |
| **RB-BOW-PROTO** | 54.9 | 63.7 | 69.4 | 73.6 | 75.4 | 75.8 | 75.5 |
| **RB-BOW-SVM** | 54.9 | 61.9 | 68.4 | 73.5 | 76.1 | 76.7 | 76.5 |
| **BOW-PROTO** | 51.2 | 54.4 | 55.2 | 61.1 | 63.6 | 68.7 | 69.2 |
| **RB-WAVG-BERT** | 50.3 | 51.6 | 53.4 | 58.8 | 69.5 | 78.3 | 81.0 |
| **AVG-BERT** | 51.1 | 51.9 | 53.2 | 60.2 | 65.6 | 72.2 | 72.8 |
| **RA-SVM** | 52.2 | 52.4 | 55.6 | 58.1 | 62.7 | 62.2 | 63.2 |
| **SVM** | 51.2 | 51.3 | 52.8 | 55.3 | 62.1 | 69 | 70.9 |
| **RA-CNN** | * | * | 54.6 | 53 | 74.9 | 82.8 | 83.8 |
| **CNN** | * | * | 53.7 | 53 | 69.9 | 79.8 | 81.1 |
| **ULMFiT** | * | * | * | 53.7 | 59.9 | 65.9 | 69.3 |

Table 3: ASRS test set results.

methods both in their rationale and non-rationale versions. We note that the RA-CNN baseline takes a similar approach to ours in the sense that it also combines pre-training (pre-trained word embeddings) with rationales. The reasons for one method being better than the other on a particular dataset could be related to the compatibility between the pre-training corpus and the test dataset or sensitivity to hyperparameter tuning (we did not do any tuning on the ASRS dev set).

## 7 Conclusions

In this work, we addressed an important challenge in supervised machine learning, namely the dependency on large amounts of labeled training data. We demonstrated that substantial performance gains in low-shot text classification can be obtained by combining unsupervised pre-training with annotator rationales across various methods. To this end, we presented two novel methods that together provide strong results on a range of train set sizes. We performed experiments using various baselines, data sizes and ablations to help understand what works best for varying amounts of available training data. Most notably, we showed that simple bag-of-words methods with pre-trained word embeddings work best for very small train sets, while more complex methods based on pre-trained language models excel when more data is available.

## References

Muhammad Arshad Ul Abedin, Vincent Ng, and Latifur Khan. 2011. Learning cause identifiers from annotator rationales. In *IJCAI*, pages 1758–1763.

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

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and

Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *EACL*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional lstm. In *CoNLL*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *EMNLP*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*.

Manali Sharma and Mustafa Bilgic. 2018. Learning with rationales for document classification. *Machine Learning*, 107(5):797–824.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *ACL*.

Michael Tepper, Heather L Evans, Fei Xia, and Meliha Yetisgen-Yildiz. 2013. Modeling annotator rationales with application to pneumonia classification. In *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. 2017. Generative and discriminative text classification with recurrent neural networks. *arXiv preprint arXiv:1703.01898*.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. In *NAACL*.

Omar Zaidan, Jason Eisner, and Christine Piatko. 2007. Using annotator rationales to improve machine learning for text categorization. In *NAACL*.

Ye Zhang, Iain Marshall, and Byron C Wallace. 2016. Rationale-augmented convolutional neural networks for text classification. In *EMNLP*.