Investigating the Working of Text Classifiers

Devendra Singh Sachan♠,△, Manzil Zaheer♠, Ruslan Salakhutdinov♠
♠School of Computer Science, Carnegie Mellon University
△Petuum, Inc
Pittsburgh, PA, USA
{dsachan, manzilz, rsalakhu}@andrew.cmu.edu

Abstract

Text classification is one of the most widely studied tasks in natural language processing. Motivated by the principle of compositionality, large multilayer neural network models have been employed for this task in an attempt to effectively utilize the constituent expressions. Almost all of the reported work train large networks using discriminative approaches, which come with a caveat of no proper capacity control, as they tend to latch on to any signal that may not generalize. Using various recent state-of-the-art approaches for text classification, we explore whether these models actually learn to compose the meaning of the sentences or still just focus on some keywords or lexicons for classifying the document. To test our hypothesis, we carefully construct datasets where the training and test splits have no direct overlap of such lexicons, but overall language structure would be similar. We study various text classifiers and observe that there is a big performance drop on these datasets. Finally, we show that even simple models with our proposed regularization techniques, which disincentivize focusing on key lexicons, can substantially improve classification accuracy.

1 Introduction

Text classification is one of the fundamental tasks in natural language processing (NLP) in which the objective is to categorize text documents into one of the predefined classes. This task has a lot of applications such as topic classification of news articles, sentiment analysis of reviews, email filtering, etc. Text classification still remains a significant challenge in language understanding as we need to encode the intrinsic grammatical relations between sentences in the semantic meaning of a document. This is especially crucial for sentiment classification because relations like "contrast" and "cause" can have great influence on determining the meaning and the overall polarity of a document.

Traditionally, for text classification, bag-of-words model (Harris, 1954) was used to represent a document. This simple approach uses the term frequency as features followed by a classifier such as Naïve Bayes (McCallum and Nigam, 1998) or Support Vector Machine (SVM) (Joachims, 1998). One drawback of this approach is that it ignores the word order and grammatical structure. It also suffers from data sparsity problem when the training set's size is small but has shown to give good results when size is not an issue (Wang and Manning, 2012). Before applying these methods, feature selection is typically carried out to reduce the effective vocabulary size by removing the noisy features (Manning et al., 2008). A key property of these linear classifiers is that they assign high weights to some class label specific keywords, which are also known as lexicons.

The next generation of approaches includes neural networks that have shown to outperform bag-of-words models in text classification tasks (Kim, 2014; Johnson and Zhang, 2016). These methods typically use multiple layers of Convolutional Neural Network (CNN) (Lecun et al., 1998) and/or Long Short Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997). The motivation of using these complex neural network approaches for classification tasks comes from the principle of compositionality (Frege, 1948), which states that the meaning of a longer expression (e.g. a sentence or a document) depends on

Proceedings of the 27th International Conference on Computational Linguistics, pages 2120–2131
Santa Fe, New Mexico, USA, August 20-26, 2018.

the meaning of its constituents. It is believed that lower layers of the network learn representations of words or phrases, and as we move up the layers more complex expressions are represented (Peters et al., 2018).

However, we believe that these state-of-the-art text classification techniques do not actually follow this mechanism, despite the motivations. Like any discriminative approach which can pick up any signal, it appears that these techniques most likely still learn and rely heavily on key lexical items or phrases and just use these lexicons to classify the document, which might not generalize to new documents.

To test our hypothesis, we first construct datasets (Section 3) where the training and test splits have no direct overlap of such key lexicons while taking care of class imbalance, although remaining language structure remains the same. Such dataset split also occurs in many real-world classification problems. For example, in the case of scientific documents, with the discovery of a new phenomenon/law/theorem, a new keyword is born. The document containing the new word would still belong to one of the existing classes, such as optics/biochemistry/discrete math, and a text classifier should be able to correctly classify the given document as the language structure would remain the same.

We study the performance of various text classifiers on this new training-test split and compare performance results with the commonly used random training-test split of such a dataset (Section 5). We observe that there is a big performance gap in current approaches between the two splits. We further show that even simple regularization techniques of replacing key lexicons with random embeddings can improve performance on the training-test split where there is no overlap of such keywords. We also present a novel approach based on the gradient of word embeddings that leads to further improvement on such dataset split (Section 4). We also provide two large-scale text classification datasets which contain both the random split and lexicon based split.

We report additional experimental results by modeling the above problem as that of unsupervised domain adaptation where the data distribution of training domain and test domain are different (Ganin et al., 2016; Zhang et al., 2017). Even such domain adaptation approaches do not yield better results than our proposed regularization techniques (Section 5).

2 Background and Related Work

In this section, we first discuss work related to neural network approaches for text classification followed by a discussion on recent domain adaptation techniques which can be thought of as an alternative solution for our problem.

2.1 Neural Network Approaches

Recent research on text classification tasks makes use of neural networks which has shown to outperform methods based on the bag-of-words model. These approaches take distributed representation of words as input which is also known as word vectors (Mikolov et al., 2013). These word vectors can be learned either using skip-gram (Mikolov et al., 2013) or Glove (Pennington et al., 2014) methods. One remarkable property of these vectors is that they learn the semantic relationships between words such that in the embedding space, semantically similar words are closer together. To learn document embeddings from these word vectors, Le and Mikolov (2014) use distributed bag-of-words (DBOW) approach also known as paragraph vectors while Joulin et al. (2017) computes the average of the word and subword vectors of a document to train a linear classifier.

To extract sentence level features for text classification task, Kim (2014) uses shallow CNN with max-pooling on top of pre-trained word vectors. They also observe that learning task-specific vectors through fine-tuning leads to better classification accuracy. Similarly, LSTM network pre-trained using language model parameters or sequence autoencoder parameters is used by Dai and Le (2015) for various text classification tasks. Recently, it was shown by Johnson and Zhang (2015a) that a CNN with dynamic max pooling layer can effectively use the word order structure when trained using one-hot encoding representation of input words. They also learn multi-view region embeddings through semi-supervised learning and incorporate them as additional features to further improve the model's performance (Johnson and Zhang, 2015b). Similarly, they also perform semi-supervised experiments using a simplified LSTM

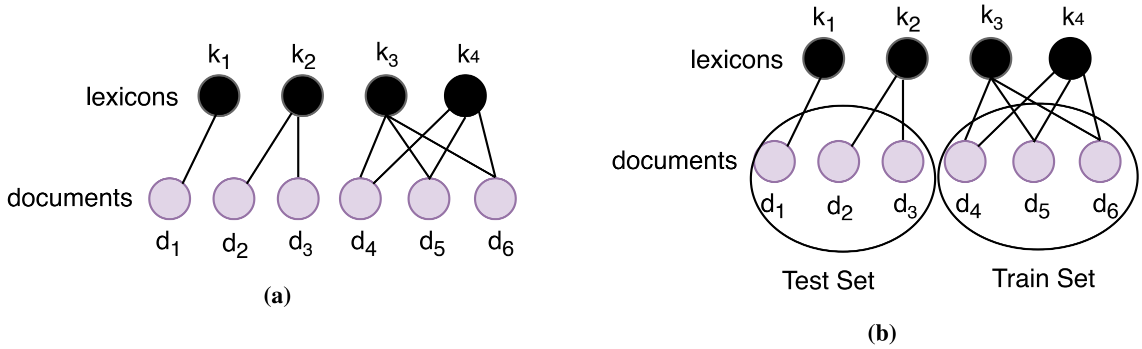


Figure 1: Schematic diagram illustrating the methodology of lexicon dataset construction. In (a), we construct a bipartite, undirected graph between the identified keywords ($k_1, \ldots, k_4$) and all the documents ($d_1, \ldots, d_6$). We identify all the connected components in this graph. In (b), all the documents belonging to the largest connected component (d_4, d_5, d_6) are selected as the training set while those documents belonging to all the other connected components (d_1, d_2, d_3) are selected as the test set.

model which also takes one-hot encoding of words as input (Johnson and Zhang, 2016).

2.2 Domain Adaptation

Next, we discuss the recent works applicable to domain adaptation of text classifiers. In unsupervised domain adaptation, the labeled instances from training data are considered as the source domain while unlabeled instances from test data are considered as the target domain. In this paper, it is assumed that there exists covariate shift between the two domains such that the conditional distribution of class label remains the same in both the domains while the marginal distribution of instances may differ (Shimodaira, 2000; Bickel et al., 2007). To address the problem of covariate shift, attempts have been made by adversarially training the encoder so as to make document representation domain invariant (Ganin et al., 2016).

Another approach to domain adaptation is multi-task learning as it can help improve the performance of text classifiers on lexicon dataset (Caruana, 1997). It is assumed that by learning related tasks in parallel while using a shared representation of document encoder can improve generalization by inducing proper inductive biases.

3 Lexicon Dataset Construction

As mentioned previously, to test our hypothesis whether the text classifiers just focus on key lexicons, we consider three text corpora: IMDB reviews, its standard subset (Maas et al., 2011), and Arxiv abstracts. For each dataset, we construct two versions: random split version and lexicon split version. In the random split version, selection of training and test examples is done by random sampling. The ratio of test to training examples is kept approximately the same for both the random and lexicon version of each dataset. In the lexicon split version, we make sure that key lexicons in training and test examples do not overlap while taking care of class imbalance. Below, we provide stepwise details of the approach used for the construction of lexicon version of the datasets.

1. **Identification of important label specific lexicons:** We begin by extracting *tf-idf* weighted, unigram to five-gram word features for all the documents. Using thus obtained feature vectors as input, we train a multinomial Naïve Bayes classifier for predicting the class on the entire dataset (both training and test examples). From the trained Naïve Bayes classifier, we extract the feature weights. Next, we rescale the feature weights by dividing all the feature values by the corresponding minimum value for that feature across all the classes. Lastly, to identify lexicons, we select the top-*k* features with the maximum weight for every class. We set the value of *k* to be 1500 for IMDB datasets and 150 for Arxiv abstracts dataset. We also experiment with Logistic Regression and SVM classifiers but observe that multinomial Naïve Bayes selects the most diverse set of features.

	cs.AI	cs.IR	cs.CV	cs.RO
Train	reinforcement learning logic programming markov decision processes probabilistic inference	information retrieval recommender systems collaborative filtering recommendation systems	computer vision image segmentation deep convolutional neural super resolution	motion planning humanoid robot dynamic environments aerial vehicle
Test	bayesian networks graphical models constraint satisfaction	matrix factorization social networks search engines	pose estimation optical flow sparse representation	multi robot simultaneous localization extended kalman filter

(a) Arxiv abstracts

	most-positive	positive	neutral	negative	most-negative
Train	absolutely amazing awesome experience fantastic movie	completely satisfied good experience great movie	average movie is just ok moderate movie	disappointed scope for improvement movie sucks	a huge let down movie was awful very frustrating
Test	outstanding performance very impressive	nice experience movie is good	satisfactory movie avg performance	not a good no plot in movie	worst experience pretty bad

(b) IMDB reviews

Table 1: Example lexicons for training and test set of Arxiv abstracts and IMDB reviews datasets. The column headers in both the tables indicate the names of various classes in these datasets. For Arxiv abstracts dataset, the details of all the class names can be found in the URL: `https://arxiv.org`

2. **Creation of lexicons-documents graph:** Since each label specific lexicon term can occur in multiple documents, and one document can contain many such lexicons, in order to create a disjoint partition of lexicons and documents into training/test sets, it is natural to represent documents and lexicons using a graph-based structure. Therefore, for each class, we create an undirected graph whose nodes are lexicons and documents respectively. If a lexicon occurs in a document, we create an edge in the graph between the two corresponding nodes. The resulting graph is bipartite, as edges only exist between the lexicon nodes and the document nodes but not among each other (Figure 1a).

3. **Graph partitioning to generate training/test splits:** We compute all the connected components of the lexicons-documents graph. We then identify the largest connected component, i.e. one which contains the maximum number of nodes and select all the documents in it as the training set.[1] The remaining documents from all the other connected components are selected as the test set (Figure 1b). If the ratio of the number of test to training documents is below a cutoff threshold, we repeat the above steps by gradually selecting fewer top-*k* features (i.e. smaller set of lexicons). Some example lexicons for training/test splits of Arxiv abstracts and IMDB reviews datasets is shown in Table 1.

We want to emphasize that after the creation of training and test sets, the majority of the words appear in both the sets for the lexicon version.

4 Methods

In this section, we will first briefly describe a simple neural network for text classification on which our proposed regularization methods are based.

Let the vocabulary size be V and embedding dimension be D. Our network consists of an embedding layer ($E \in \mathbb{R}^{V \times D}$), single layer bidirectional LSTM (BiLSTM) (Schuster and Paliwal, 1997), a pooling layer, and finally a linear layer with softmax function for classification. The sequence of word ids ($w_t, t \in [1, T]$) are given as input to the embedding layer which maps them to dense vectors. The forward LSTM and backward LSTM of a BiLSTM processes these input sequence of word vectors in forward and reverse directions respectively. The hidden state of forward LSTM and backward LSTM is concatenated at every time-step and are passed to the pooling layer which computes the maximum value over time

[1] We also tried spectral partitioning of lexicons-documents graph but it didn't generate balanced graph partitions.

dimension to obtain the representation of the input sequence (Conneau et al., 2017). We train the model by minimizing the cross-entropy loss.

While training various neural network models on both random and lexicon splits of ACL IMDB dataset (Maas et al., 2011), we observed that on the lexicon split, the network is easily able to learn the training data distribution in 1-2 epochs. These models also overfit on the lexicon split as the accuracy gap between the training and test set widens up. We hypothesize that this happens because, during training step, the model is able to memorize common label-specific keywords occurring in the training set. During evaluation, when the model is not able to spot such keywords in the test set, as it contains non-overlapping partition of keywords from the training set, the performance degrades quite rapidly as compared to the random split version. Therefore, motivated by these observations, we propose two methods which attempt to prevent neural networks from memorizing the word order structure in keywords by introducing more randomness in them.

4.1 Keyword Anonymization

In order to prevent the model from memorizing keyword specific rules and thus learning degenerate representations, we introduce more randomness in our training data split (Hermann et al., 2015). In the first step, we identify some keywords with high scores using a supervised classifier trained using bag-of-words as features. The first step is similar to the first step of data construction, with the important distinction that now we only operate on training set as test set is assumed to be unknown during the training phase. In the second step, we anonymize the corpus by replacing a single word selected at random from the occurrences of such keyword phrases with a placeholder word ‘`ANON`’. In the third step, we assign a random word embedding during model training for every occurrence of the placeholder word ‘`ANON`’. These modified word embeddings are given as input to the neural network encoder.

In this way, we corrupt the information present in keywords by introducing some form of random noise in the data. This is one of the ways to regularize model training. We later show that this regularization forces the network to learn context-aware text representations which gives a significant gain in classification accuracy. Motivated by the effectiveness of this method, we now propose an end-to-end version as the next approach.

4.2 Adaptive Word Dropout

This approach is based on embedding layer’s gradient which is computed at every optimization step. The motivation of this approach is based on the observation that key lexicon terms have a higher norm of the embedding gradient. In this method, we first compute the word embedding gradient ($\nabla E \in \mathbb{R}^{V \times D}$) during the backward step. In the second step, for every word we maintain the running average ($A(w)$) of the L2-norm of this gradient. This average is computed with respect to the number of completed optimization steps.

$$\tilde{A}(w) = \tilde{A}(w) + \sqrt{\sum_{j=1}^{D} \nabla E_{ij}^2}$$
$$A(w) = \frac{1}{\text{optim\_steps}} \tilde{A}(w)$$

In the third step, we compute the dropout probability of every word as:

$$P_d(w) = 1 - \sqrt{\frac{t}{A(w)}}$$

In the above equation, t is a hyperparameter (typical values of t are 10^{-3}, 10^{-4}). We apply dropout on the words before the word embedding layer with probability $P_d(w)$ (Srivastava et al., 2014). We also apply variational dropout (Gal and Ghahramani, 2016) on the output of word embedding layer.

	Arxiv abstracts				IMDB reviews				ACL IMDB			
	lexicons		random		lexicons		random		lexicons		random	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
c	127	127	127	127	5	5	5	5	2	2	2	2
l	162	140	153	153	299	252	278	281	290	254	234	228
N	668K	438K	664K	443K	1.48M	1.07M	1.49M	1.07M	23K	26K	25K	25K
V	297K	238K	301K	244K	590K	509K	601K	505K	75K	78K	75K	74K

Table 2: This table shows the dataset summary statistics. c: Number of classes, l: Average length of a sentence, N: Dataset size, V: Vocabulary size

5 Experiments

We now present empirical studies in order to establish that (i) many text classifiers resort to just identifying key lexicons and thus poorly performing on specially crafted dataset splits (Section 5.3), and (ii) simple regularization techniques which disincentivize focusing on key lexicons can significantly boost performance (Section 5.4).

5.1 Dataset Description

To illustrate aforementioned claims, we evaluate on an existing dataset for binary sentiment classification and also collected two more large-scale text classification datasets. The details of these three datasets are described below:

- **ACL IMDB**: This is a popular benchmark dataset (Maas et al., 2011) of IMDB movie reviews for coarse-grained sentiment analysis in which the task is to determine if a review belongs to the positive class or to the negative class. The original random split version of this dataset contains an equal number of highly positive and highly negative reviews. To construct its lexicon based version, we apply our approach to the combined training and test splits of this dataset.

- **IMDB reviews**: This is a much bigger version of the IMDB movie reviews dataset in which the task is to do fine-grained sentiment analysis. We collect more than 2.5 million reviews from IMDB website and partition them into five classes based on their ratings out of 10. These classes are *most-negative*, *negative*, *neutral*, *positive*, and *most-positive*.

- **Arxiv abstracts**: In this, the task is to do multiclass topic classification. We construct this dataset by collecting more than 1 million abstracts of scientific papers from "*arxiv.org*". Each paper has one primary category such as cs.AI, stat.ML, etc. which we use as its class label. We selected those primary categories which have at least 500 papers. In order to extract text data, we use the title and abstract of each paper. In both the Arxiv abstracts and IMDB reviews dataset, we keep the ratio of the number of instances in the test set to that of training set as 0.6.

We pre-process the data by converting all the text to lowercase followed by Penn Treebank style tokenization. An overall summary of the training/test sets of lexicon and randomized splits is provided in Table 2.

5.2 Experimental Setup

In this section, we will discuss our experimental setup. For simple baseline comparisons, we conduct experiments with methods which take bag-of-words as input feature representations. Specifically, we compute *tf-idf* weighted *n-gram* features for each document to train multinomial Naïve Bayes and Logistic Regression classifiers. For neural network models, we have a common experimental setup based on Pytorch framework (Paszke et al., 2017) for all the datasets unless specified otherwise. Due to computational constraints, we did not perform extensive hyperparameter tuning for the methods considered.

We select most common 80K words for model training. We initialize the embedding layer parameters for all the models using 300-dimensional pre-trained embeddings. These embeddings are trained using skip-gram approach (Mikolov et al., 2013) on the combined training and test set.[2] The embeddings of words which are not present in these vectors are uniformly initialized. The hidden state of LSTM and BiLSTM has 1024 and 512 dimensions respectively. For training of CNN models, we follow the settings mentioned in Kim (2014). We perform model training using mini-batch stochastic gradient descent with a batch size of 150. Optimization is performed using Adam Optimizer (Kingma and Ba, 2014) with default parameter settings. To train LSTM models, we perform backpropagation for 250 timesteps on IMDB datasets and 150 timesteps on Arxiv abstracts dataset. For model regularization, we apply dropout (Srivastava et al., 2014) with probability 0.5 to the input and output of LSTM. In order to prevent gradient explosion problem, we perform gradient clipping (Pascanu et al., 2013) by constraining the norm of the gradient to be less than 5.

5.3 Results

In order to estimate the difficulty level on both the lexicon and random splits version of a dataset, we do experiments with a wide range of popular approaches and show their results in Table 3. It can be observed from the results that for each dataset, there is a big performance gap between the random split and lexicon split for all the models. In this section, we will analyze these performance results. During our analysis, we will mostly focus on classifier's performance on the lexicon version of a dataset. For a detailed analysis of the performance of a method on the random version, we encourage the reader to read the associated paper.

5.3.1 Bag-of-Words Model

Here, we analyze the performance of methods which takes bag-of-words representation as input. We observe that the simple generative classifier of Naïve Bayes performs poorly as compared to other approaches on both Arxiv abstracts and IMDB reviews dataset, while its results are competitive on ACL IMDB dataset. Because of its relatively low scores, we don't experiment with more complex generative models. In contrast, a simple discriminative classifier such as Logisitic Regression performs significantly better than Naïve Bayes on the random version for all the datasets. However, on lexicon version, it's performance is quite low when compared to neural network methods. Due to this, the accuracy gap for this method is largest between both the dataset versions. These low scores of above classifiers on lexicon version can be explained owing to the conditional independence assumption among the input features. During training step, bag-of-words based models assign high weights to class-specific keywords. When this trained model is not able to spot such discriminative keywords in the test set due to strict no overlap condition, the performance of these methods degrades.

5.3.2 Simple Word Embedding based Methods

Next, we analyze the results of classifiers which are based on simple linear or nonlinear transformation of word embeddings. We train 300-dimensional document vectors (Le and Mikolov, 2014) using standard training settings for DBOW model.[3] Similarly, we use the recommended training setting for FastText method (Joulin et al., 2017). These methods further improve the lexicon results in both the IMDB datasets. Although DBOW and FastText methods don't strictly make use of word order in a sentence, improved performance suggests that they leverage the semantic properties of the embedding space as the document vectors are also learned in the same geometrical space as the word vectors.

5.3.3 Convolutional and Recurrent Networks

As discussed previously (Section 2), it has been shown that both CNN and LSTM models can learn effective text representations. We observe that these networks show much better performance on both the random and lexicon version as compared to the above-discussed methods. We also note that the use of a

[2] We use *word2vec* tool from `https://code.google.com/p/word2vec`

[3] We use the existing implementation from the open-source gensim toolkit.

Model	**Arxiv abstracts**			**IMDB reviews**			**ACL IMDB**		
	random	lexicon	Δ	random	lexicon	Δ	random	lexicon	Δ
Naïve Bayes	57.6	40.4	17.2	54.9	41.6	13.3	89.8	79.3	10.5
Logistic Regression	65.2	40.6	24.6	60.0	41.9	18.1	90.5	80.6	9.9
DocVec	62.6	38.9	23.7	55.9	44.4	11.5	88.6	82.8	5.8
FastText	66.8	43.8	23.0	57.9	45.1	12.8	88.5	80.5	8.0
Deep Sets	55.7	37.9	17.8	52.0	46.8	5.2	88.2	79.6	8.6
LSTM	65.0	45.3	19.7	64.6	48.6	16.0	90.6	81.9	8.7
BiLSTM	**67.8**	46.5	21.3	**64.8**	48.9	18.9	**91.4**	83.1	8.3
CNN-MaxPool	65.8	46.0	19.8	50.5	44.1	6.4	89.6	80.5	9.1
CNN-DynMaxPool	66.6	45.5	21.1	52.1	41.5	10.6	90.0	80.9	9.1
Adv-Training	–	45.1	–	–	47.3	–	–	82.2	–
Multi-task Learning	–	43.5	–	–	44.8	–	–	78.7	–
LSTM-Anon	67.2	48.2	19.0	62.6	50.3	12.3	89.0	83.1	5.9
BiLSTM-Anon	67.5	48.7	18.8	62.8	51.4	11.4	89.7	84.1	5.6
Adaptive Dropout	67.2	**49.1**	18.1	64.0	**52.6**	11.4	91.2	**86.0**	5.2

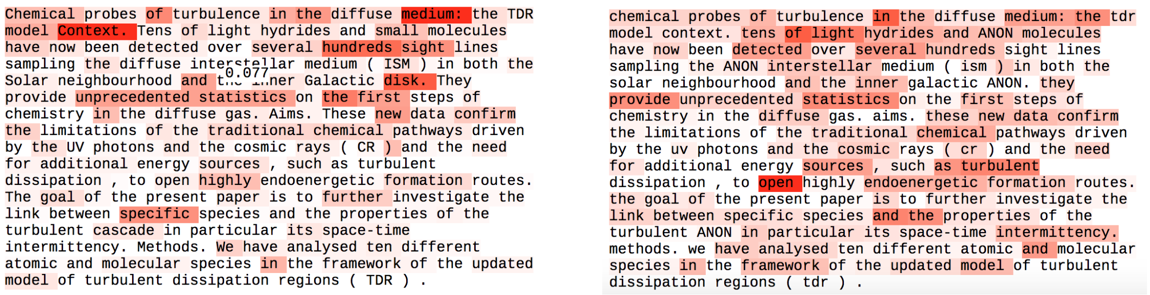
Table 3: Classification accuracy of various models on random and lexicon based version of each dataset. We also show the accuracy difference (Δ) between these two results. **Naïve Bayes**: *n-gram* feature extraction using *tf-idf* weighting followed by Naïve Bayes classifier. **Logistic Regression**: *n-gram* feature extraction using *tf-idf* weighting followed by Logistic Regression classifier. **DocVec**: Document vectors trained using DBOW model (Le and Mikolov, 2014). **FastText**: Average of the word and subword embeddings (Joulin et al., 2017). **Deep Sets** (Zaheer et al., 2017): two layer MLP on top of word embeddings with max pooling layer. **LSTM**: Word level LSTM as a document encoder in which hidden state of the last time-step is used for classification. **BiLSTM**: Word level bidirectional LSTM as a document encoder in which forward LSTM and backward LSTM hidden states are concatenated followed by max pooling layer (Conneau et al., 2017). **CNN-MaxPool**: CNN with max pooling layer (Kim, 2014). **CNN-DynMaxPool**: CNN with dynamic max pooling layer. For details on dynamic max pooling, we request the reader to refer to Johnson and Zhang (2015a). **Adv-Training**: Adversarial training of encoder is done to fool the discriminator and make the representation of training and test instances domain invariant. **Multi-task Learning**: A shared BiLSTM encoder is used to perform joint training of text classifier, denoising autoencoder, and adversarial training. **LSTM-Anon**: LSTM encoder applied to the anonymized training data. **BiLSTM-Anon**: BiLSTM encoder applied to the anonymized training data. **Adaptive Dropout**: BiLSTM encoder applied after embedding gradient-based adaptive word dropout.

more complex model such as BiLSTM gives performance improvement in both random and lexicon splits for all the datasets.

Although there is an improvement in classification accuracy, still there is a large drop in performance from random to lexicon split. For the best performing approach of BiLSTM model, accuracy difference between random and lexicon split of ACL IMDB, IMDB reviews, and Arxiv abstracts dataset is 8.3%, 18.9%, and 21.3% respectively. In order to narrow down this performance gap, there is a need for approaches which can learn more robust context-based representations so that the performance on documents containing new unseen key phrases can be improved.

5.3.4 Domain Adaptation Methods

For the lexicon split, one can consider that the training and test data distributions are different, and thus model this as an instance of unsupervised domain adaptation. We evaluate two different approaches for such domain adaptation: adversarial training and multi-task learning.



Chemical probes of turbulence in the diffuse medium: the TDR model Context. Tens of light hydrides and small molecules have now been detected over several hundreds sight lines sampling the diffuse interstellar medium (ISM) in both the Solar neighbourhood and t 0.077 ner Galactic disk. They provide unprecedented statistics on the first steps of chemistry in the diffuse gas. Aims. These new data confirm the limitations of the traditional chemical pathways driven by the UV photons and the cosmic rays (CR) and the need for additional energy sources , such as turbulent dissipation , to open highly endoenergetic formation routes. The goal of the present paper is to further investigate the link between specific species and the properties of the turbulent cascade in particular its space-time intermittency. Methods. We have analysed ten different atomic and molecular species in the framework of the updated model of turbulent dissipation regions (TDR) .

(a) without word anonymization

chemical probes of turbulence in the diffuse medium: the tdr model context. tens of light hydrides and ANON molecules have now been detected over several hundreds sight lines sampling the ANON interstellar medium (ism) in both the solar neighbourhood and the inner galactic ANON. they provide unprecedented statistics on the first steps of chemistry in the diffuse gas. aims. these new data confirm the limitations of the traditional chemical pathways driven by the uv photons and the cosmic rays (cr) and the need for additional energy sources , such as turbulent dissipation , to open highly endoenergetic formation routes. the goal of the present paper is to further investigate the link between specific species and the properties of the turbulent ANON in particular its space-time intermittency. methods. we have analysed ten different atomic and molecular species in the framework of the updated model of turbulent dissipation regions (tdr) .

(b) with word anonymization

Figure 2: This attention heat map shows the change in hidden state activations of BiLSTM encoder when some label specific keywords are anonymized for an example instance from Arxiv abstracts dataset whose label is *astro-ph.GA*. In Figure 2a, the trained classifier makes an incorrect prediction as it can be seen that the model is performing a keyword-based classification. In Figure 2b with word anonymization, the attention span also involves other context words and the prediction is correct.

To address lexical/covariance shift in lexicon dataset split, we adversarially train BiLSTM model with the objective that the resulting document representation will be discriminative for the text classification task while indiscriminative to the training/test domain classification. To perform such domain-invariant adversarial training, we include a discriminator module which consists of three feed-forward layers. To learn the model parameters, we follow the training method of Ganin et al. (2016), but instead of gradient reversal, we feed the interchanged labels of the domains to fool the discriminator. From the results, we observe that adversarial training of encoder hurts the performance on lexicon datasets.

In multi-task learning, the various tasks which were trained jointly are text classification, denoising autoencoder (Hill et al., 2016), and adversarial training. In all these tasks, we share the embedding layer and BiLSTM encoder layer. Denoising autoencoder was trained on the entire dataset using the strategy described in Lample et al. (2017). From the results, we observe that multi-task learning doesn't help and it further hurts the overall classification accuracy.

We didn't experiment with these domain adaptation approaches on the random split of the dataset, as it is assumed that it contains the same distribution of lexicons in the training and test set. Also, as the initial results were not promising on lexicon dataset, we didn't try more complex approaches.

5.4 Our Methods

BiLSTM encoder applied to anonymized training data with random embedding substitution performs around 2% better on Arxiv abstracts dataset, 2.5% better on IMDB reviews dataset and gives around 1% improvement on ACL IMDB lexicon dataset. This shows that keyword anonymization with random embedding substitution can be a good strategy for model regularization in case of lexicon-based split. From a qualitative perspective, we show in Figure 2 the change of hidden state activations for the BiLSTM encoder after the data is anonymized.

Our approach of adaptive word dropout performs 2.6% better on Arxiv abstracts, 3.6% better on IMDB reviews and gives around 2.8% improvement on ACL IMDB lexicon datasets. We also note that this approach leads to an only marginal drop in accuracy for the random split version of the datasets. One of the reasons why this method is more effective is that word dropout partially masks some lexical terms in the training set thereby lowering the variance of the fitted model.

6 Conclusion

Recently, multilayer neural network models have gained wide popularity for text classification tasks due to their much better performance than traditional bag-of-words based approaches. It is widely believed that this happens as neural networks can effectively utilize the word order structure present in documents. But, a potential drawback is that since all neural network approaches are discriminative, they tend to identify key signals in the training data which may not generalize to test data. In this work, we investigate whether these neural network models actually learn to compose the meaning of sentences or just use discriminative keywords.

To test the generalization ability of different state-of-the-art text classifiers, we construct hard datasets where the training and test splits have no direct overlap of lexicons. Our experiments with popular text classifiers show that there is a large drop in test classification accuracy between random and lexicon splits of these datasets. We show that simple regularization techniques such as keyword anonymization can substantially improve the performance of text classifiers. We also observe that adaptive word dropout method which is based on embedding layer's gradient can further improve accuracy and thus reduce the gap between the two dataset splits.

Acknowledgments

This work was supported by a generous research funding from CMU, MCDS students grant. We would also like to thank the anonymous reviewers for giving us their valuable feedback which helped to improve the paper.

References

[Bickel et al.2007] Steffen Bickel, Michael Brückner, and Tobias Scheffer. 2007. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 81–88, New York, NY, USA. ACM.

[Caruana1997] Rich Caruana. 1997. Multitask learning. *Mach. Learn.*, 28(1):41–75, July.

[Conneau et al.2017] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680. Association for Computational Linguistics.

[Dai and Le2015] Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 3079–3087, Cambridge, MA, USA. MIT Press.

[Frege1948] Gottlob Frege. 1948. Sense and reference. *The philosophical review*, 57(3):209–230.

[Gal and Ghahramani2016] Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1019–1027. Curran Associates, Inc.

[Ganin et al.2016] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17(1):2096–2030, January.

[Harris1954] Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

[Hermann et al.2015] Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada.*

[Hill et al.2016] Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1367–1377. Association for Computational Linguistics.

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November.

[Joachims1998] Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, ECML'98, pages 137–142, Berlin, Heidelberg. Springer-Verlag.

[Johnson and Zhang2015a] Rie Johnson and Tong Zhang. 2015a. Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 103–112, Denver, Colorado, May–June. Association for Computational Linguistics.

[Johnson and Zhang2015b] Rie Johnson and Tong Zhang. 2015b. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 919–927, Cambridge, MA, USA. MIT Press.

[Johnson and Zhang2016] Rie Johnson and Tong Zhang. 2016. Supervised and semi-supervised text categorization using lstm for region embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 526–534. JMLR.org.

[Joulin et al.2017] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain, April. Association for Computational Linguistics.

[Kim2014] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October. Association for Computational Linguistics.

[Kingma and Ba2014] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

[Lample et al.2017] Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

[Le and Mikolov2014] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1188–II–1196. JMLR.org.

[Lecun et al.1998] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov.

[Maas et al.2011] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

[Manning et al.2008] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

[McCallum and Nigam1998] Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive bayes text classification. In *Proceedings of AAAI-98, Workshop on Learning for Text Categorization*, pages 41–48. AAAI Press.

[Mikolov et al.2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

[Pascanu et al.2013] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages III–1310–III–1318. JMLR.org.

[Paszke et al.2017] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October. Association for Computational Linguistics.

[Peters et al.2018] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

[Schuster and Paliwal1997] M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681, November.

[Shimodaira2000] Hidetoshi Shimodaira. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227 – 244.

[Srivastava et al.2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January.

[Wang and Manning2012] Sida Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 90–94, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Zaheer et al.2017] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. 2017. Deep sets. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3391–3401. Curran Associates, Inc.

[Zhang et al.2017] Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. 2017. Aspect-augmented adversarial networks for domain adaptation. *Transactions of the Association for Computational Linguistics*, 5:515–528.