Convolutional Neural Network Baseline Replication for Sentence Classification

Matthew Etchells
260680753

Brendon Keirle
260685377

Jonathan Pearce
260672004

***Abstract*—Sentence classification is concerned with a variety of tasks in natural language processing, including the prediction of opinion in text. In the past standard supervised learning models such as Naive Bayes, Support Vector Machine and Logistic Regression. In recent years there has been a shift towards the use of deep learning approaches to these tasks. In this project we look at a publication by Yoon Kim outlining the use of CNNs for sentence classification tasks across several datasets. Kim uses a baseline model of a simple CNN for comparison with the more complex approaches proposed. It may be important to compare novel deep learning approaches to baselines that use standard supervised learning models. We look at the extension of standard supervised learning models to these same datasets, and found that these models can consistently outperform the simple CNN baseline.**

CONTENTS

I. INTRODUCTION

Sentence classification describes a range of text classification tasks including subjectivity and sentiment analysis. Several data-sets have been used to evaluate the performance of machine learning models on sentence classification tasks. Extensive research has gone into the development of standard machine learning models for these tasks, including Support Vector Machine (SVM), Naive Bayes (NB) and Logistic Regression (LR) classifiers [1]. In more recent years, research interested has shifted towards deep learning approaches to these tasks, making use of complex neural network architectures.

Among deep learning research in the realm of sentence classification is that by Yoon Kim in a publication that focused on the development of Convolutional Neural Network (CNN) models [2]. CNNs were originally proposed to deal with image data [3], but have since transitioned into popular use for Natural Language Processing (NLP) tasks as well [4] [5]. Kim proposed the use of CNN's taking advantage of dimensionality reduction done with word vectors, using dedicated software such as Word2Vec [6]. The use of word vectors lends well to CNNs in which sparse matrices are transformed into a network layer of dense, trainable representations. Word vectors project word features into a multi-dimensional space, and the similarity of words based on their context or meaning is related to the distance between their vectors [7]. The use of word vectors allows the capture of relationships between features that cannot easily be captured by standard approaches to feature extraction such as one-hot-encoding [8] or Term Frequency Inverse Document Frequency (TFIDF) [9]. Although Kim achieves high accuracies with their CNN and word vector approach, comparison is made only to a baseline CNN model and not to "shallow" learning models that had dominated sentence classification tasks before deep learning made its way into the realm.

In this project we analyzed the extent to which SVM,

NB and LR models could compete with our implementation of the simple CNN baseline used by Yoon Kim in their validation set accuracy. Among the 6 data-sets selected by Yoon Kim, we experimented with the 3 that had binary classes and relatively few training examples. We found that SVM and NB models matched or outperformed the CNN baseline in all three data-sets analyzed, and competed with the more complex CNN and word vector model reported by Kim in one of the three.

II. Related Work

Sentence classification was originally studied in the context of standard supervised learning algorithms [10]. In recent years deep learning models have gained popularity for these tasks [11]. A more popular deep learning architecture for NLP tasks is the LSTM [12]. LSTM's are well suited to text data since the construction of these networks naturally deal with sequential data such as sentences. Work done by Fan [13] similarly experimented with naive bayes used for sentiment classification. Fan also demonstrated the benefits of alternate approaches such as maximum entropy [14].

III. Dataset and Setup

A. Datasets

Dataset	c	N	| V |
MR	2	10662	18344
Subj	2	10000	20916
MPQA	2	10606	6195

TABLE I: Summary statistics for datasets after text processing. c: Number of classes. N: Dataset size. | V |: Vocabulary size

The MR, SUBJ, and MPQA datasets contain 10662, 10000, and 10606 training examples respectively. Each training point in the MR dataset is a single sentence extracted from a movie review. The predicted binary labels of each datapoint in the SUBJ dataset represent whether the input sentence is subjective or objective in nature. The MPQA Opinion Corpus contains news articles from a wide variety of news sources manually annotated for opinions and other private states (i.e., beliefs, emotions, sentiments, speculations, etc.).

B. Text Processing

Every sentence in each dataset was cleaned and formatted in the same manner. All punctuation was replaced with a space character except for exclamation and question marks as they could potentially assist in the classification process. Each sentence was converted to lower case and all excess white space characters were removed.

Further processing was required for the convolutional neural network experiments. We converted each sentence to a one hot vector where each word in a dataset vocabulary was assigned an integer to represent it, thus each sentence containing l words was now represented numerically by l integers. The next step was to make each sentence the same length, this ensured there would be no dimensionality issues in the convolutional layer of the neural network. Every one-hot vector was zero padded (from the front and back evenly) so that each sentence was of length n. Here $n = l_{max}+8$, where l_{max} was the length of the longest sentence in the dataset and 8 ensured that the largest filter size (5) would completely traverse every word in the sentence.

IV. Models

A. Convolutional Neural Networks

Convolutional neural networks [15] are a type of neural network originally designed to work well with image data [3]. As opposed to regular feed forward neural networks, convolutional neural network layers are capable of accepting three dimensional input (tensors), and outputting transformed three dimensional data. Using local receptive fields convolutional neural networks are able to find distinct image features and pooling/subsampling helps reduce the network dimensionality, layer by layer. Convolutional neural networks are able to effectively handle the high dimensions associated with image data, exploit the 2D topology of pixels in images and can be designed to be invariant to certain expected variations between images such as translations, rotations, and illumination differences [16]. Since their inception, convolutional neural networks have been found to be effective with more than just image tasks. They are becoming the standard in audio tasks [17], [18] and very competitive in text processing tasks [19], [20], [21]. The paper we have based this project on is a clear example of the latter.

The first model we implemented in this project was a replication of the baseline model from [22], 'CNN-rand'. Our replicated model was developed using Keras [23] with a Tensorflow [24] backend. The CNN-rand architecture contains four network layers. Layer 1 is a word embedding layer, that accepts an n x 1 one-hot vector representation of a sentence of length n and produces an n x 300 array of word embeddings. This array is then

reshaped to a 3-dimensional tensor of size n x 300 x 1. At the start of network training each dimension value of the word embeddings is randomly initialized using a uniform distribution $U[-0.245, 0.245]$, these word embeddings are then modified during training. Layer 2 is the convolution layer, it consists of three separate 2-dimensional filters of size: 3 x 300, 4 x 300 and 5 x 300. The ReLu activation function [25] is applied to all outputs from layer 2. Nonlinear functions such as ReLu, sigmoid, and hyperbolic tangent play an important role in CNNs.

Although the ReLu function is actually a piecewise linear, it enables neural networks to represent complex nonlinear functions. 100 activation maps are produced for each filter size, therefore this layer produces 3 tensors of size (n-2) x 1 x 100, (n-3) x 1 x 100 and (n-4) x 1 x 100. Layer 3 is for subsampling, it uses max pooling overtime to reduce each tensor to size 1 x 1 x 100. Max pooling overtime is important for dealing with variable sentence lengths across a dataset. Next we concatenate and flatten these three tensors to form one column vector with dimension 300 x 1. Our fourth layer is a fully connected layer which accepts a 300 x 1 vector as input, and uses a softmax activation function to output a 2 x 1 vector representing the model's belief over each class label. Additionally, Layer 4 also utilizes two regularization techniques to help prevent the network from overfitting. The first is dropout [26] with a probability p=0.5, the second is enforcing a l_2-norm constraint of 3 on the weight vectors in Layer 4. For training we used stochastic gradient descent with shuffled mini-batches of size 50 and the adaDelta optimizer [27] with default parameters.

B. Standard Learning Models

MNB, SVM and LR models were evaluated on accuracy using 10 fold cross validation. For each model and dataset, 10 trials were performed and accuracy results were averaged. Different trials were performed to include varying parameters: (1) the number of features, (2) feature encoding (i.e. one-hot encoding, feature counts or TFIDF) and (3) extents of regularization. Regularization for MNB models was achieved by changing the extent of laplace smoothing. SVM and LR models varied the extent of L2 regularization by changing the coefficient of the penalty term.

MNB uses a simple probabilistic linear model, taking advantage of bayes' rule and the assumption of conditional independence between features [28] [29]. It has long shown success sentence classification and other NLP tasks, especially in the case of smaller datasets.

SVM is a non-probabilistic model that aims to divide sample points of different classes with the largest margin to create a decision boundary [30] [31]. SVMs have proven effective in sentence classification tasks as well [10], and were generally accepted as the method of choice for classification tasks prior to the emergence of neural network models.

LR is another linear model that can be interpreted as probabilistic and has history of success on sentence classification tasks.

The encoding of features was done in 3 ways: (1) one-hot-encoding, (2) word counts and (3) TFIDF. These 3 feature representations were compared across MNB, SVM and LR for each of the datasets.

V. Results

Model	**MR**	**Subj**	**MPQA**
CNN-rand (Kim '15)	76.1	89.6	83.4
CNN-rand (Replication)	74.4	89.0	83.3
MNB	**78.9**	**92.6**	83.1
SVM	78.3	91.6	**86.3**
LR	77.7	90.5	84.3

TABLE II: Accuracy of Kim's CNN-rand baseline against our CNN-rand replication and standard model baselines

For each of the 3 datasets we were able to implement a random CNN baseline with similar accuracies to Kim's, and standard learning models to outperform it.

Table (II) shows the mean accuracy scores of CNN-rand, MNB, SVM and LR models on each of the datasets. MNB scored the highest accuracy on the MR and Subj datasets, when using TFIDF features, and laplace smoothing with $\alpha = 1$. SVM had the highest accuracy on the MPQA dataset.

Model	**MR**	**Subj**	**MPQA**
CNN-rand (Replication)	632	566	398
MNB	2.29	1.91	0.60
SVM	3.35	3.42	0.86
LR	8.72	3.40	0.79

TABLE III: Training time in seconds comparison of our CNN-rand replication and standard model baselines

Table (III) compares training times of the various models. Standard machine learning models were able to achieve similar or greater accuracies as the random CNN baseline at a small fraction of the computational

cost of training.

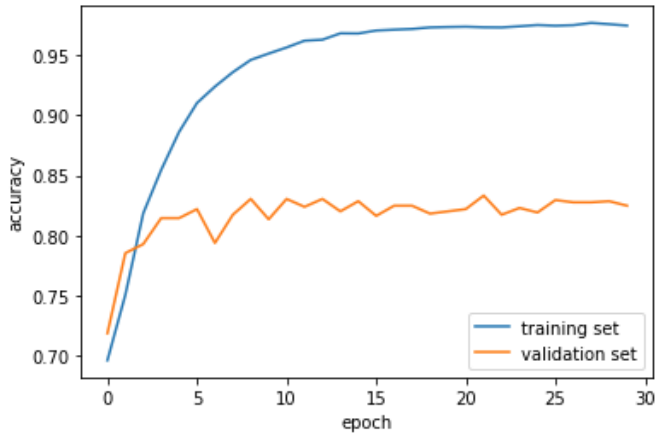

Fig. 1: CNN-rand replication accuracy on training and validation set

Figure 1 shows the accuracy of the CNN-rand on the training and validation sets across 30 epochs of training on the MPQA dataset. Validation accuracy plateaus after approximately 4 epochs while training accuracy continues to grow, indicating that CNN-rand quickly over fits on the small MPQA dataset.

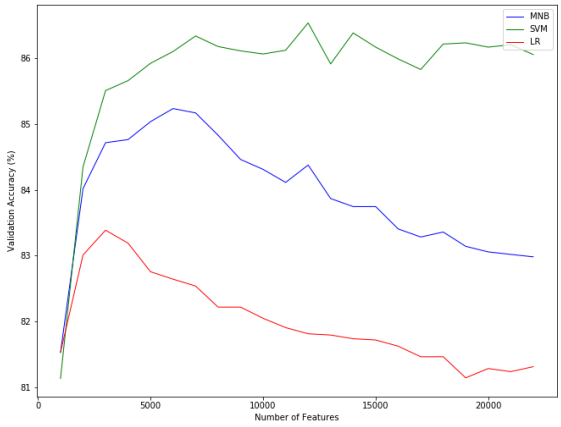

Fig. 2: Number of Features against Accuracy for standard Learners on MPQA Dataset

Figure (2) and (3) look at the validation accuracies of each of the standard learning models with increasing numbers of features for the MR and MPQA datsets respectively. Features were iteratively added based on their frequencies in the training data, and validation accuracy was assessed with 10 fold cross validation. Figure (2) shows that MNB and SVM models reach high accuracy and either improve or are stable with increasing numbers of features. LR on the otherhand loses validation accuracy with added features, indicating a potential overfit. On the otherhand, Figure (3) shows that SVM is the only model that does not seem to saturate and overfit the training data on the small MPQA dataset whereas MNB and LR both seem to lose validation accuracy with added features.

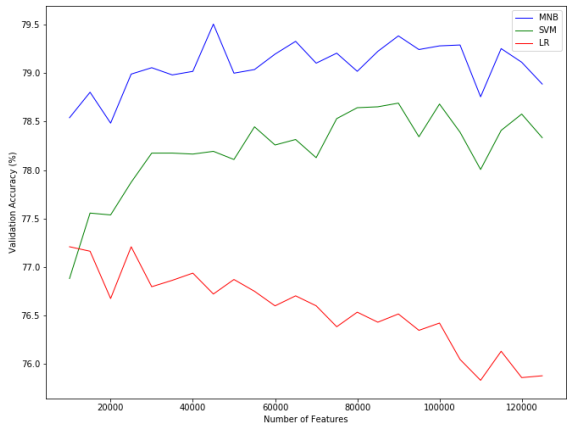

Fig. 3: Number of Features against Accuracy for standard Learners on MR Dataset

VI. Discussion

A. Replication Correctness

Overall our work to replicate Kim's CNN-rand architecture and training process was relatively successful. For each of the three datasets we were able to come within 2% of the accuracy reported in [22]. In his paper Kim does an adequate job of explaining his experimental process, the model architecture, as well as including references to close variants of the model architecture [32] [33]. Withstanding this, there are still a few key points missing that led to potential sources of error in our replication process. The first detail missing from [22] was the method utilized in pre-processing each sentence of text. We assumed that since this information was excluded that Kim did not perform any significant or out of the ordinary text processing techniques and assumed similar processes would yield similar results. Comparing our vocabulary sizes in Table 1 to those reported in [22], it is clear that our pre-processing technique did not replicate that of Kim's exactly, however is potentially very close. Although our results demonstrate that this

difference in pre-processing did not have a huge impact on model performance, it was disappointing to have a potential source of replication error introduced in the very first experimental step, as this error was then carried through the entire experiment. The next source of potential error in our baseline replication was from padding the sentences to equal length. In [22] there is no mention of the technique used to pad sentences. Due to this lack of information we took our padding technique from [32], Kim does cite this paper which made us more confident about the possibility that he employed the same technique however with just [22] there is no way to know for sure whether our assumption about the padding process is correct. In general Kim outlined the network setup well, explaining each layer and it's parameters clearly as well as how each pair of layers were connected. The only potential source of error we found in replicating the CNN-rand architecture was in the random initialization of the word embeddings, specifically the dimensionality of the word embeddings and the distribution used to initialize the values. Kim provides no explicit details about how this was done in [22], however there are comments that allowed us to make relatively safe assumptions in order to solve these two issues. With regards to the dimensionality of the word embeddings, we noted that page 3 of [22] Kim discusses the word2vec vectors [6] used in his more sophisticated models and that these vectors have dimensionality 300. We assumed that it would make sense for the randomly initialized word embeddings to be of the same length, and therefore gave them dimensionality 300. To solve for what distribution to use for randomly initializing the word embeddings we looked to a detail on page 5 of [22]. Kim discusses that when randomly initializing words not in word2vec there were slight improvements in model performance when each dimensions' value was sampled from a uniform distribution $[-a, a]$, with a such that the randomly initialized word embeddings had the same variance as the word2vec vectors. This was the only discussion in the paper about word embedding initialization technique and therefore it is the process we followed for CNN-rand. [34] provided us with the variance of the word2vec vectors trained on the 100 billion word Google News corpus, from that we were able to obtain $a = 0.245$. The remainder of the network architecture was clearly explained in [22]. The final source of error in our replication process came from choosing a method to stop the network training. In [22] Kim discusses that he employed the adaDelta optimizer to train the network, however provides no criteria for stopping the training procedure. We decided to select the model at epoch k if the validation accuracy of the network at epoch $\{k + 1, ..., k + 5\}$ was less than the validation accuracy of the network at epoch k. We found it surprising that there was no information provided in [22] about their training stop criteria, especially after discovering that our CNN-rand replication model greatly overfit the training data, as demonstrated in Figure 3. Given the relatively small datasets and massive number of trainable parameters in the CNN-rand model (most of which were for the word embeddings), it's not surprising that the CNN-rand model quickly overfits the training data. This overfitting made the need for the training stop criteria crucial, as it became unclear whether training in [22] was stopped as soon as the network began to overfit the data or whether a criteria more similar to the one we selected was used. We believe overfitting would effect the more advanced models in [22] such as CNN-static much less than it did with CNN-rand. The advanced models' word embeddings are always initialized by the word2vec vectors from the 100 billion word Google News corpus. Even when the word2vec vectors remained the same, like in CNN-static, the model was able to outperform CNN-rand by over 3% on each dataset used in our experiments (Table 2 of [22]). Therefore it is clear that word embeddings trained on a massive general corpus of text are still much more reliable than domain specific word embeddings trained on a relatively small corpus of data. Due to this we suspect that CNN-static, CNN-non-static and CNN-multichannel in [22] did not suffer from the same extreme amount of overfitting that we observed with our CNN-rand replication model. This belief is strengthened by the fact that this concern was not addressed in a follow up to Kim's work [35]. It is because of these reasons and the fact that CNN-rand was not the focus of his paper, that Kim did not address this overfitting issue and potentially why he did not include his criteria for stopping network training.

B. Performance of MNB, SVM and LR Classifiers

The MNB model achieved highest accuracy on the MR and SUBJ datasets, and SVM achieved highest accuracy on the MPQA dataset. The only case in which the random CNN baseline outperforms the standard model baselines is on the MPQA dataset where the baseline's accuracy exceeds only MNB's. However, the SVM achieved the highest accuracy in this dataset, while MNB had the highest accuracy on the other two. One possible explanation for the success of MNB contrasted with the poor performance of the CNN may be related

to using small datasets. MNB offers a relatively simple linear model compared to Logistic Regression and SVM with high bias and low variance. It thus lends well to small datasets that may be susceptible to overfitting by more complex models, as seen in the case of the random CNN. As a result of the high bias intrinsic to MNB, the validation accuracy of MNB models was seemingly unaffected by increasing numbers of feature counts. The performance of the MNB model seems to make the most of very few features and training examples without suffering from overfitting. On the other hand, LR models struggled in all cases compared to MNB and SVM. The reasoning behind this may be in contrast to MNB, such that LR is a relatively lower bias and higher variance model. As a result LR models struggle with such small numbers of features and training examples. The validation accuracy of LR models tended to decrease with increasing numbers of features, indicating overfitting of the model. We note that LR accuracy may have improved with In the case of the MPQA dataset, the SVM model performed best. MNB accuracy was similar for approximately 5000 features or fewer. The MNB model indeed seemed to overfit the data beyond this point: the dataset contained 6195 unique words, and so beyond this threshold the majority of features considered would have been bigrams with low frequency in the training set. Overfitting may have had to do with the use of many features with single occurrences in the training set that did not well generalize to validation data.

VII. Conclusion

We successfully implemented a random CNN baseline that gave similar results to that of [22] for sentence classification tasks. As well, we trained MNB, SVM and LR classifiers that were able to outperform the CNN baseline on the 3 datasets that we analyzed. The small datasets lended well to higher bias, lower variance models and thus MNB and SVM were favored. The random CNN baseline on the other hand suffered from overfitting. standard machine learning models such as MNB and SVM may serve as higher accuracy baselines for sentence classification tasks than do simple CNN models, and at a fraction of the computational cost.

VIII. Statement of Contributions

Jonathan worked on implementation, data collection/analysis and the write up portion for CNN baselines. Matthew and Brendon worked on implementation, data collection/analysis and the write up portion for MNB, SVM and LR baseline models.

REFERENCES

[1] Tony Mullen and Nigel Collier. Sentiment analysis using support vector machines with diverse information sources. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 412–418, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[2] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, Apr 1980.

[3] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324, 1998.

[4] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *CoRR*, abs/1708.02709, 2017.

[5] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of CNN and RNN for natural language processing. *CoRR*, abs/1702.01923, 2017.

[6] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[7] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. From word embeddings to document distances. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 957–966. JMLR.org, 2015.

[8] David Harris. *Digital design and computer architecture*. Morgan Kaufmann, Waltham, MA, 2013.

[9] Shahzad Qaiser and Ramsha Ali. Text mining: Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181, 07 2018.

[10] Thorsten Joachims. 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, 1998. Springer-Verlag.

[11] Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis : A survey. *CoRR*, abs/1801.07883, 2018.

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

[13] Weiguo Fan, Linda Wallace, Stephanie Rich, and Zhongju Zhang. Tapping the power of text mining. *Communications of the ACM*, 49:76–82, 09 2006.

[14] Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, March 1996.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[16] Karel Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. *CoRR*, abs/1411.5908, 2014.

[17] Karol J. Piczak. Environmental sound classification with convolutional neural networks. *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2015.

[18] Justin Salamon and Juan Pablo Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24:279–283, 2017.

[19] Cícero Nogueira dos Santos and Maíra A. de C. Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, 2014.

[20] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc., 2015.

[21] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2042–2050. Curran Associates, Inc., 2014.

[22] Yoon Kim. Convolutional neural networks for sentence classification. In *EMNLP*, 2014.

[23] François Chollet et al. Keras. https://keras.io, 2015.

[24] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[25] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, USA, 2010. Omnipress.

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

[27] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[28] Pattern classification and scene analysis. richard o. duda , peter e. hart. *The Library Quarterly*, 44(3):258–259, 1974.

[29] David D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *Proceedings of the 10th European Conference on Machine Learning*, ECML '98, pages 4–15, London, UK, UK, 1998. Springer-Verlag.

[30] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

[31] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg, 1995.

[32] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *CoRR*, abs/1103.0398, 2011.
[33] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *CoRR*, abs/1404.2188, 2014.
[34] Yu Su and Xifeng Yan. Cross-domain semantic parsing via paraphrasing. *CoRR*, abs/1704.05974, 2017.
[35] Ye Zhang and Byron C. Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *CoRR*, abs/1510.03820, 2015.