MiniProject 4 - Reproducibility

Sam Cleland (260675996), Meko Deng (260637443) and Matthew Lesko-Krleza (260692352)

***Abstract*— The work attempted during this project was to re-implement baselines proposed in the paper *Sentiment Analysis for Text Classification (T4)* by Yoon Kim [1]. The models that were re-implemented as baselines in the paper are CNN-Static and CNN-Rand. We also implemented models of our own that competed with the re-implemented models in the form of an LSTM, a hybridized SVM with Naive Bayes like features and a normal SVM. Extensive hyperparameter tuning was done in the form of cross validation and search for optimal parameters and evaluation of performance was done on a test set from several of the datasets Yoon Kim had used in his paper. The datasets we used include MR, CR and SST. Lastly, if a test set was not given by a dataset, we performed an 90/10 train test split and used the 10% to test performance. If not provided then we used the training and test sets provided. Pre-processing is done in the form of regular expressions for tokenizing meaningful elements of a sentence. Despite the lack of complexity, both SVMs seemed to provide competitive performance to the reproduced CNN models.**

I. Introduction

Sentiment Analysis for Text Classification (T4) explores multiple designs of training CNNs using word vectors that were trained with unsupervised learning. The word vectors that were used were outputted by word2vec, a two-layer neural net that outputs word vectors that can be used as initial weights in a CNN or LSTM. The differences between the models proposed by Kim were the way that some of them were initialized (such as randomly or with these word vectors) and the way that these weights changed (either the word2vec vectors were left the same or dynamically changed from back-propagation). Minimal hyper-parameter tuning was done on their proposed models and validation was done by using a 90 to 10 training to test split if the dataset did not have a given test set. The results reflected by his experiment showed that the models performed well with respect to other state of the art models at the time with only 1 convolutional layer covering the word vectors (CNN static and non-static) generated from word2vec.

The proposed approach for reproducing the results from the experiment were to implement some of the CNN models they had used (CNN-Rand and CNN-Static) as baselines and then create 3 new models that would offer similar or better performance, in the form of an LSTM, hybridized Naive Bayes SVM and regular SVM. 3 of the datasets that Yoon Kim had used in his experiments were used for training and testing. The 3 datasets that were used were Movie Review, Customer Review and SST-2. If the dataset did not have an already configured test set, a 90 training 10 test split was used, similar to how Kim had done train/test splits. Preprocessing was done in the form of regular expression tokenization and lemmatization. Despite not as much complexity, the Linear SVM provides competitive performance across all the datasets for the reproduced models, as well as the NBSVM slightly behind. This suggests that opting for less complex models with heavy preprocessing and appropriate feature extraction mechanisms can perform just as well as higher complex models for sentiment classification.

A. Preliminaries

Neural Networks (NN) are systems that learn how to classify the input they are given correctly. This is done by stacking multiple models in the form of hidden units and training them jointly. They can also be referred to as Deep Neural Networks (DNN) if they have one or more hidden units.

Convolution is a mathematical operation on two functions (f and g) to produce a third function that expresses how the shape of one is modified by the other, can be visualized as sliding one function across the other and the output being the multiplication of the two functions at each time step.

Convolutional Layer is a layer that has a filter with specific dimensions that "slide"/convolve with batches of the data according to the filter's size. This allows for the layer to extract patterns from the input.

Convolutional Neural Networks (CNN) are deep neural networks that use convolutional layers.

Epochs are the number of iterations during neural network training.

Activation Functions are used to indicate if a neuron should be "activated or not" based on a non linear transformation of the output at the previous layer. Popular transformations include the sigmoid function ($\sigma = 1/(1 + e^{-x})$) or Rectified Linear Unit (ReLU $= \max(0, x)$).

Dropout regularization associates a probability of setting a particular node to 0 for each node in a particular layer, a technique useful to prevent overfitting [2].

Max Pooling is the idea of taking the maximum of different elements within a layer, similar to the convolutional filter where we slide across the input with an n by n pooling filter and perform some non linear operation with that n by n patch of the image. This is done to reduce dimensionality of hidden layers, resulting in higher level feature abstraction.

Batch Normalization is a technique to improve training efficiency by normalizing data for each training batch at each layer of the network. The mean and variance are independently calculated for each dimension [3].

Softmax maps vectors of values to a vector of probabilities. Useful when applying a softmax to the last layer of

a NN as it is compatible with cross entropy loss and max likelihood.

Cross Entropy Loss Function is a measure of how many bits an encoding will require if we use information from the neural network's output compared to the actual predicted values. Cross-entropy loss increases as the predicted probability diverges from the actual label.

Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent for scaling the learning rate for each parameter based on statistics of the history of gradient descent. For example, if a parameter wasn't updated very much in a previous iteration, increase the update strength for that particular parameter for the next iteration [4].

Early Stopping refers to terminating a training routine once validation error has reached a minimum to avoid overfitting.

Law of Diminishing Returns is a concept that refers to a point in which a benefit gained (in NN context, model accuracy) is less than the effort invested (in NN context, training time)

Naive Bayes Support Vector Machine is a type of support vector machine that has naive bayes log count ratios as features [5].

Recurrent Neural Network or RNN is a class of neural networks where connections between nodes form a directed graph along a temporal sequence [6].

Long Short-Term Memory or LSTM is an RNN architecture model. The major difference in design is the addition of cells composed of an input gate, an output gate and forget gate. The LSTM is a solution for exploiding or vanishing weights in RNNs [7].

Word embeddings/vectors are used to convert sentences into vectors of scores that represent a particular word. This is done primarily for a dimensionality reduction on the feature space, wouldn't have to use a 1-hot encoding to represent the occurrence of a word or combination of words.

Word2Vec is a two-layer neural net that processes text that takes as input a text corpus and outputs a set of vectors that represent feature vectors for words in that corpus. More simply, Word2Vec translates a given text into a numerical form that can be consumed by Neural Networks.

II. Related Work

A sophisticated and high performing approach includes that of an Attention-Gated CNN [8]. The model achieves accuracy scores of 86.4, 81.7, and 87.2 on CR, MR, and SST-2 respectively. The model generates attention weights from the feature's context windows at different sizes by using specialized convolution encoders, thereby extracting and enhancing the influence of important features in predicting the sentence's category.

III. Dataset and Setup

Yoon Kim's CNN models were tested on various datasets, four of which was of interest in our case:

	MR	CR	SST
Size	10662	3944	10754

TABLE I
Dataset Sizes

- **MR**: Movie reviews with one sentence per review. [9]
- **SST-2**: Stanford Sentiment Treebank 2 - a finely-tuned extension of MR but with train/dev/test splits provided. The neutral reviews are removed and binary labels added for sentiment. This dataset was taken from [10].
- **CR**: Customer reviews of an Apex DVD player, a Canon camera, a jukebox, a Nikon camera, and a Nokia phone [11].

The datasets sizes can be viewed under Table I. The size is the number of datapoints.

Upon acquiring these datasets, we also performed minimal text preprocessing in order to clean up the text. This process involves tokenizing a given sentence, then lemmatizing the tokens.

A. Pre-trained Word Vectors

Additionally, in order to recreate the CNN-Static model discussed in Yoon's paper [1], we made use of the pubicly available word2vec vectors that were trained on 100 billion words from Google News. Though there were many options for vector dimensions, we opted for using a vector width of 300 as mentionned in Yoon's paper [1]. This method allows initializing word vectors which compensates for the absence of a large supervised training set.

IV. Proposed Approach

We begin by reproducing two baseline models proposed on Yoon Kim's paper - that is - the CNN-Rand model and the CNN-Static model. The CNN-Rand model initializes all word vectors are randomly initialized and modified during training. This means that the word2vec data were not used to retrieve word vectors. The CNN-Static model uses the pre-trained vectors with all the words fetched from the word2vec dataset and the unknown words randomly initialized - all of these words are kept static and only the other parameters of the model are learned. We were able to reproduce similar results to Yoon's CNN-Rand model by mimicking the described approach in the paper. A summary of the guidelines followed as as follows:

- ReLU activation
- Filter windows of 3,4,5 with 100 feature maps each.
- Dropout Rate of 0.5
- l2 constraint(s) of 4
- mini-batch size of 50

Each convolution pool are then downsized using a Global Max Pooling layer. An overview of the reconstructed CNN-Rand and CNN-Static models can be seen on Figure 1 and Figure 2 respectively. Furthermore, all models were trained using 10 epochs. This choice is further discussed in Section V.

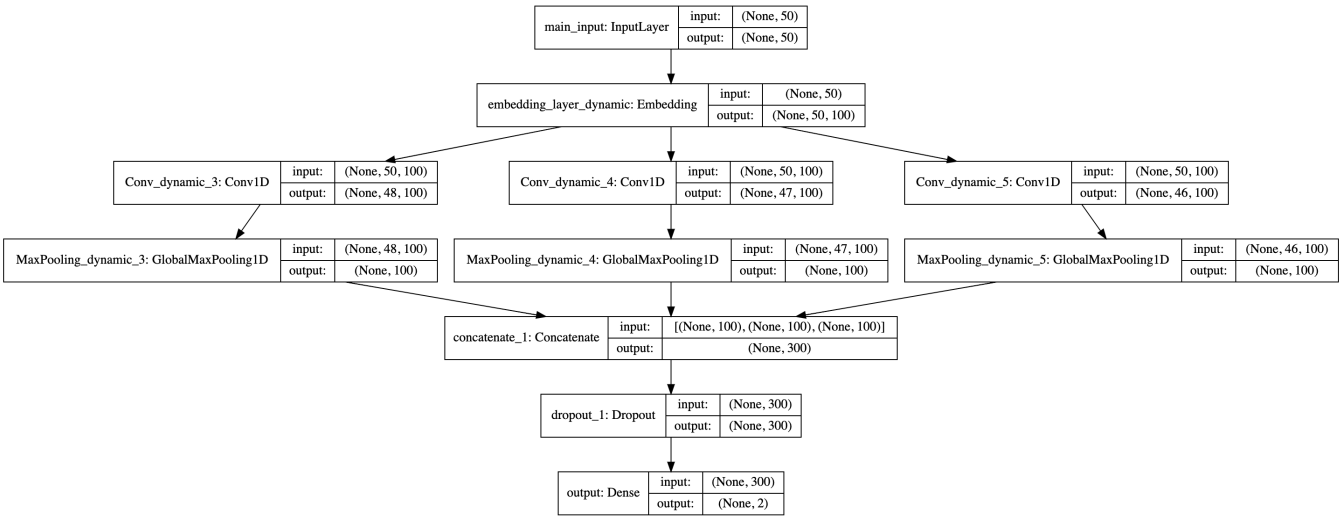

Fig. 1. Overview of the reproduced CNN-Rand Model

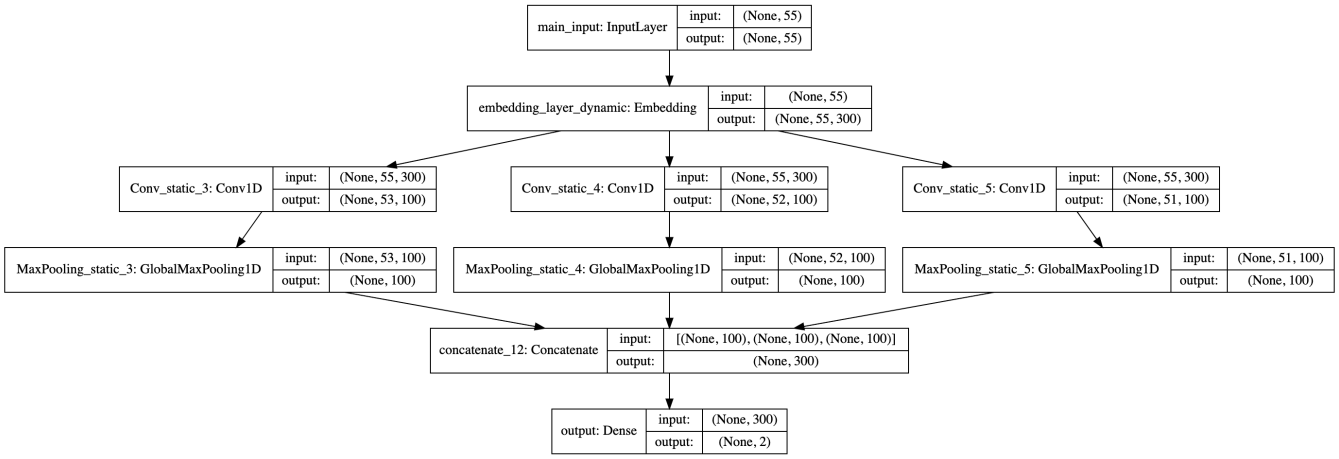

Fig. 2. Overview of the reproduced CNN-Static Model

Subsequently, we implemented three additional models in order to introduce new baselines and simple alternatives to perform Sentiment Analysis for Text Classification: Naive Bayes Support Vector Machine, Support Vector Machine and LSTM.

A. Implemented Models

1) Naive Bayes Support Vector Machine: The first model implemented to compete against the baselines was a hybridized Naive Bayes support vector machine composed of TF-IDF weighting, bi-grams and l2 regularization. Implementation of the classifier was referenced from here, also is in *References* below [12]. The main reason why a NB support vector machine felt appropriate in this context was because of the independence assumption between features from naive bayes. In this scenario, it seemed appropriate to implement this assumption because of the 1 hot encoded like features that are extracted from our feature pipeline (TF-IDF weights of unigrams and bigrams). This turned out to be not as beneficial as thought but still ended up with decent accuracy compared against the baselines.

2) Support Vector Machine: A support vector machine was implemented to compare the baseline results of the Naive Bayes support vector machine, with all of the same feature extractions as the NBSVM except with no naive bayes log feature counts.

3) Cross Validation on SVM models: Cross validation in the form of *RandomizedSearchCV* was used on both SVMs. *RandomizedSearchCV* was done to ensure that a suitable setting of parameters was found for each model, as the optimal parameters from dataset to dataset could change. The parameter spectrum consisted of testing different values of C (penalty term for how much to consider miss-classification), max iterations for convergence and the type of loss used.

4) LSTM: We decided to implement an LSTM because of its popularity and success in natural language processing.

The LSTM was trained on the CR, MR, and SST1 datasets using the same static word-to-vector encoding as the CNN-Static model. The input layer is a matrix embedding layer. The hidden layer is an LSTM with ReLU activation, and 0.5 dropout rate. The final layer is a fully connected layer with Softmax activation. For the MR and SST2 datasets, the LSTM was trained for 10 epochs, whereas for the CR dataset, the model was trained for 1 epoch. A dropout of 0.5 was chosen because it proved us useful in the last assignment. We only used one LSTM layer because we wanted to implement a simple model.

B. Libraries Used

Our models were constructed using keras, a high-level neural networks API, written in Python running on a Tensorflow backend[13] because of its simplicity in use. We also made use of the keras text preprocessing function: Tokenizer as well as python's nltk library. Furthermore, we made use of the scikit-learn library for splitting our dataset as well as hyperparameter tuning(GridSearchCV and RandomizedSearchCV), pipelining and metrics tracking. Finally, we used CometML - a platform that allows monitoring, comparing and optimizing our ML models automatically keeping track of individual experiments and results.

V. Results

A. Reproduced CNN Models

An important issue noted while training the reproduced models is their their tendencies to overfit extremely fast. This behaviour is likely due to the low number of data in our given dataset. In fact, Neural Networks in general do not perform well with a small number of data. Furthermore, this behaviour can also be explained by the fact that there is no optimizer in place to adjust and regulate the learning rate of the model. We first tested the models with a high epoch value of 50, but soon realized the models' performance decreased over time. Hence, we chose to choose a low epoch value of 10, which led to results similar to the ones obtained by Yoon.

Another issue discovered while attempting to recreate the models was the lack of information in the preprocessing of the given dataset. In fact, we speculate that this lack of information is one of the reasons why our results deviates from the original baseline model as proposed in Yoon's paper. Furthermore, some technicalities in the CNN models were not explained in detail which led to assumptions such as using either the GlobalMaxPooling1D or the MaxPooling1D library in keras. The former was chosen upon closer insepction of Yoon's paper.

1) CNN-Rand: Running the CNN-Rand model yielded unstable results due to the nature of the model, which would randomize the initial word vector. This explains the over-performance of the reproduced model in the MR (Movie Review) dataset category and the under-performance on the CR (Customer Review) dataset category as seen in Figure 1.

Models	MR	CR	SST2
CNN_rand(original)	76.1	79.8	82.7
CNN_rand(reproduced)	**83.7**	73.0	82.3
CNN_static(original)	81.0	84.7	**86.8**
CNN_static(reproduced)	79.1	**85.9**	85.0
NB_SVM	81.9	78.5	81.5
Linear_SVC	81.1	81.5	81.3
LSTM	74.9	77.2	79.6

TABLE II
COMPARISON OF ORIGINAL BASELINES AND CUSTOM BASELINES

2) CNN-Static: Because the CNN-Static model also initializes some random words in the word vector, we can still observe a slight deviation in the original results. However, this variation is significantly lower as most of the words are already present in the word vector. This behaviour can be better observed in Figure 2 by comparing the original CNN-Static model with the reproduced CNN-Static model.

3) LinearSVM and NB-SVM: The linear kernal SVM performed slightly better than the NBSVM. This could be because of the naive bayes assumption that was instilled with the features of the NBSVM. Perhaps this assumption was not very good to have within the model as there may be dependencies between features. In comparison to the reproduced models, the support vector machines performance might have to do with the size of the datasets that were used for training. CNNs tend to require a lot of data so they do not overfit, otherwise if there isn't enough data, even with dropouts between layers they can still have the tendency to be to complex and overfit the model, and if there is too much dropout then they will become too simple.

Though there are some slight deviations from the original CNN models, the general results of our reproduced model mimics the results that were obtained by Yoon Kim - that is, implementing a simple model with static vectors (CNN-Static) significantly improves the baseline model (CNN-Rand).

4) LSTM: The LSTM performed poorly, achieving the lowest validation accuracy scores on the MR and SST2 datasets. These results are likely due the complex nature of the LSTM applied on a simple Text Classification Task.

VI. Discussion and Conclusion

In this project, we first recreated two out of the four baseline models that were discussed in Yoon Kim's paper - the CNN-Rand model and the CNN-Static model. Re-implementing these models to yield the same results as the paper proved to be a challenge as the task of preprocessing the dataset was not described in the paper and some overarching model architecture was not discussed in detail. This led to some experimentation by playing with different keras library functions (Notably GlobalMaxPooling vs MaxPooling) and finding the optimal epoch number in order

to achieve the closest performance value to the original baseline models. Our results in the reimplementation had some slight deviation, but overall hovered around the same performance as Yoon's model.

Furthermore, we experimented with two additional models: SVM (Linear and NB) as well as LSTM in an attempt to gain higher results than the baselines proposed in Yoon's paper. These models did not achieve a higher result than the baselines. We noticed that complex models do not necessarily perform better at a simple task. For the LSTM, potential improvements that could be further investigated in the future include more experimentation with higher dropout rates, and adding more layers.

Given additional time, we would focus on artificially enlarging the dataset to have more data to train our complex models with - notably for the CNN and the LSTM. Furthermore, we would explore on simpler models to accomplish the simple task of Text Classification.

VII. Statement of Contributions

Matthew: Coded, trained and tested the LSTM.

Meko: Reimplemented and tested baseline CNN-Static and CNN-Rand models.

Sam: Implemented, trained and validated NBSVM and SVM models.

References

[1] Y. Kim, "Convolutional neural networks for sentence classification," 2014. [Online]. Available: https://arxiv.org/pdf/1408.5882.pdf

[2] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015. [Online]. Available: http://arxiv.org/abs/1502.03167

[4] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[5] S. Wang and C. D. Manning, "Baselines and bigrams: Simple, good sentiment and topic classification," 2012. [Online]. Available: https://www.aclweb.org/anthology/P12-2018

[6] S. Fujita and H. Nishimura, "An Evolutionary Approach to Associative Memory in Recurrent Neural Networks," *arXiv e-prints*, pp. adap–org/9 411 003, Nov 1994.

[7] H. Sak, A. Senior, and F. Beaufays, "Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition," *arXiv e-prints*, p. arXiv:1402.1128, Feb 2014.

[8] Y. Liu, L. Ji, R. Huang, T. Ming, C. Gao, and J. Zhang, "An Attention-Gated Convolutional Neural Network for Sentence Classification," *arXiv e-prints*, p. arXiv:1808.07325, Aug 2018.

[9] "Movie review data." [Online]. Available: http://www.cs.cornell.edu/people/pabo/movie-review-data/

[10] CS287, "Cs287/hw1," Feb 2016. [Online]. Available: https://github.com/CS287/HW1/tree/master/data?fbclid=IwAR3zfrc8byOQ9gKVJfL-fLgF_PuzNBo2y7hINTLYVWEk8aWdvBVo5jI-XAc

[11] "Mpqa resources." [Online]. Available: http://mpqa.cs.pitt.edu/

[12] L. Rei, "Multiclass naive bayes svm (nb-svm) - learns a multiclass classifier based on word ngrams," April 2016. [Online]. Available: https://github.com/lrei/nbsvm

[13] F. Chollet *et al.*, "Keras," https://keras.io, 2015.