The Thirty-Second International Florida
Artificial Intelligence Research Society Conference (FLAIRS-32)

# Investigation of Maxout Activations on Convolutional Neural Networks for Big Data Text Sentiment Analysis

**Gabriel Castaneda, Paul Morris, Joseph D. Prusa, Taghi M. Khoshgoftaar**
Florida Atlantic University
gcastaneda2012@fau.edu, pmorris2012@fau.edu, jprusa@fau.edu, khoshgof@fau.edu

**Abstract**

We explore the performance of multiple maxout activation variants on the big data text sentiment analysis task using convolutional neural networks. Maxout networks have gained great success in many computer vision tasks, but there is limited work on other classification tasks. Our experiments compare ReLU, LReLU, SeLU and tanh to four maxout variants. We evaluate the effectiveness of the activation functions on five datasets, including two datasets collected from the Amazon product reviews corpus, two datasets collected from the Yelp corpus, and the Sentiment140 dataset. Throughout the experiments, we found that maxout networks are slow to train compared to the traditional activation functions. We find that on average across all datasets, ReLU's classification performance is better than any maxout activation if the number of convolutional filters is doubled. Our experiments suggest that adding more filters enhances the classification accuracy of ReLU, without affecting its comparatively low training time.

## Introduction

An activation function in a neural network is a transfer function that transforms the net input of a neuron into an output signal. The output signal is then used as an input in the next layer in the stack. The activation function introduces nonlinearities to convolutional neural networks (CNNs) (LeCun and Bengio 1995), which are desirable for multi-layer networks to detect nonlinear features. Popular activation functions include sigmoid, hyperbolic tangent (tanh) and Rectified Linear Unit (ReLU) (Nair and Hinton 2010). Deep neural networks (DNNs) are models (networks) composed of many layers that transform input data to outputs while learning increasingly higher-level features. There is a lack of consensus on how to select a good activation function for a DNN, and a specific function may not be suitable for all applications. Since an activation function is generally applied to the outputs of all neurons, its computational complexity will contribute heavily to the overall execution time (Liew, Khalil-Hani, and Bakhteri 2016). DNNs have successfully utilized sigmoid units, but sigmoid activation functions suffer from gradient saturation. For this reason, different activation functions have been proposed for neural network training. ReLU has become the most commonly used activation for CNNs, yielding significant performance improvements in multiple domains.

CNNs were originally intended for computer vision tasks, being inspired by connections in the visual cortex; however, they have been successfully applied to natural language processing tasks (Poria, Cambria, and Gelbukh 2015). Typically, the first layer of the network converts words in sentences to word vectors by table lookup. The word vectors are either trained as part of CNN training, or fixed to those learned by some other method from an additional large corpus (Johnson and Zhang 2014). When working with sequences of words, convolutions allow the extraction of local features around each word.

The maxout unit (Goodfellow et al. 2013) selects the maximum value within a group of different feature maps and is usually combined with dropout (Srivastava et al. 2014) which is widely used to regularize deep networks to prevent overfitting. This technique randomly drops units or connections to prevent units from co-adapting; dropout has been shown to improve classification accuracy in some computer vision tasks (Cai, Shi, and Liu 2013). Maxout chooses the max of *n* copies of each feature in a network. The simplest case of maxout is the Max-Feature-Map (MFM) (Wu et al. 2015), where *n*=2. In the past four years, variants of maxout have been tested on benchmark datasets, and have been used in face and speech recognition tasks.

Most of the comparisons between maxout and other activation functions only report a single performance metric, ignore network size, and only report accuracy on a single dataset, with no training time or memory use analysis. Further, it is unclear whether marginal performance gains with maxout are due to the activation function or simply an increase (2x) in the number of convolutional filters versus ReLU networks. In this work, we evaluated multiple activation functions for DNNs applied to sentiment analysis. The main contributions herein can be summarized as follows:

- An evaluation of four maxout functions and comparison to popular activation functions like tanh, ReLU, LReLU and SeLU. To the best of our knowledge, this is the first study to evaluate multiple maxout variants and standard activations for big data text sentiment analysis using statistical tests.
- A comparison of training time is presented on the evaluated activation functions.
- An evaluation of whether marginal performance gains with maxout are due to the activation function, or simply an increase (2x) in the number of convolutional filters versus ReLU networks.
- An evaluation of whether maxout methods tend to converge faster and if there is a significant performance difference between these methods and standard activations.

The remainder of this paper is organized as follows. In the second section, we describe the related work on activation function evaluation for sentiment analysis. The third section describes the evaluated activation functions and datasets. The experimental methodology is presented in the fourth section. Results and analysis are provided in the fifth section. Conclusions with some directions for future work are provided in the last section.

## Related Work

Most prior work focuses on proposing new activation functions, but few studies have compared different activation functions for big data text sentiment analysis. Also, there are few comparisons between maxout and traditional activation functions. Most of the comparisons do not report the details of their network to indicate whether an increased number of filters was accounted for in the experiment and only report accuracy on a single dataset.

Maxout is employed as part of deep learning architectures and tested against the Mixed National Institute of Standards and Technology (MNIST), the Canadian Institute for Advanced Research (CIFAR-10) and CIFAR-100 benchmark datasets, but it is not compared against other activation functions using the same deep network architecture and hyperparameters on big data. It is not clear if maxout enhances the overall accuracy on the tested datasets, or if any other activation function has the same effect.

The maxout activation was shown to be effective in speech recognition tasks (Zhang et al. 2014) but it has not been widely tested on sentiment analysis. Jebbara and Cimiano used the maxout activation in their CNN portion of a hybrid architecture consisting of a recurrent neural network stacked on top of a CNN (Jebbara and Cimiano 2017). A maxout layer was also implemented in the Siamese bidirectional Long Short-Term Memory (LSTM) network proposed in (Baziotis, Pelekis, and Doulkeridis 2017). The maxout layer was selected as it amplifies the effects of dropout. The output of the maxout layer is connected to a softmax layer which outputs a probability distribution over all classes. In (Njikam and Zhao 2016), the Rectified Hyperbolic Secant (ReSech) activation function was proposed and evaluated on MNIST, CIFAR-10, CIFAR-100 and the Pang and Lee's movie review datasets. The results suggest that ReSech units are expected to produce similar or better results compared to ReLU units for various sentiment prediction tasks. The maxout network accuracy was only compared with the CIFAR-10 and MNIST datasets.

Goodfellow et al. investigated the catastrophic forgetting problem, testing four activation functions including maxout on MNIST and Amazon reviews datasets using two hidden layers followed by a softmax classification layer (Goodfellow et al. 2013). Their experiments showed that training with dropout is beneficial, at least on the relatively small datasets used in the paper and that the choice of activation function should always be cross-validated, if computationally feasible. Maxout in combination with dropout showed the lowest test errors on all experiments.

## Activation Functions and Datasets

In the following subsections, we introduce each evaluated activation function in our study.

### Hyperbolic Tangent

A hyperbolic tangent (tanh) function is a ratio between hyperbolic sine and cosine functions of x:

$$f(x) = \tanh = \frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{1}$$

### Rectified Units

Rectified Linear Unit (ReLU) (Nair and Hinton 2010) is defined as:

$$h(x_i) = \max(0, x_i) \tag{2}$$

where $x_i$ is the input and $h(x_i)$ is the output. The ReLU activation is the identity for positive arguments and zero otherwise.

Leaky ReLU (LReLU) (Maas, Hannun, and Ng 2013) assigns a slope to its negative input. It is defined as:

$$h(x_i) = \min(0, a_i x_i) + \max(0, x_i) \tag{3}$$

Where $a_i \in (0, 1)$ is a predefined slope.

The scaled exponential linear unit (SeLU) (Klambauer et al. 2017) is given by:

$$SeLU(x) = \lambda \begin{cases} x & if\ X > 0 \\ \alpha e^x - \alpha & if\ X \le 0 \end{cases} \quad (4)$$

where $x$ is used to indicate the input to the activation function. In (Klambauer et al. 2017), it justifies why α and λ must have the below values:

$$\begin{aligned} \propto &= 1.6732632423543772848170429916717 \\ \lambda &= 1.0507009873554804934193349852946 \end{aligned} \quad (5)$$

To ensure that the neuron activations converge automatically toward an average of 0 and a variance of 1.

### Maxout Units

The maxout unit takes as the input the output of multiple linear functions and returns the largest:

$$h(x_i) = \max_{k \in \{1,\dots,K\}} w^k \cdot x_i + b^k \quad (6)$$

In theory, maxout can approximate any convex function (Goodfellow et al. 2013), but a large number of extra parameters introduced by the $k$ linear functions of each hidden maxout unit result in large RAM storage memory cost and considerable training time, which affect the training efficiency of very deep CNNs. For our comparisons we use four variants of the maxout activation: an activation with $k = 2$ input neurons for every output (maxout 2-1), an activation with $k = 3$ input neurons for every output (maxout 3-1), an activation with $k = 6$ input neurons for every output (maxout 6-1), and a variant of maxout with $k = 3$ where the two maximum neurons are selected (maxout 3-2). These maxout variants have proven to be effective in classification tasks such as image classification (Goodfellow et al. 2013), facial recognition (Wu et al. 2015) and speech recognition (Cai, Shi, and Liu 2013).

The following subsections describe the big data category datasets employed in our experiments:

### Amazon Product

The original Amazon product review dataset was collected by (McAuley, Pandey, and Leskovec 2015). It contains product reviews and scores from 24 product categories sold on Amazon.com, including 142.8 million reviews spanning from May 1996 to July 2014. Review scores lie on an integer scale from 1 to 5. The sentiment dataset constructed from the Amazon product review data in (Heredia et al. 2016) was reused, where 2,000,000 reviews had a score greater than or equal to 4 stars and 2,000,000 reviews had a score less than or equal to 2 stars. The first group is labeled as positive sentiment while the second group is labeled as negative sentiment, creating a positive/negative sentiment dataset. A second subset here called "Amazon1M" was used with one million Amazon product reviews constructed in (Prusa and Khoshgoftaar 2017). The labels were automatically generated from the star rating of each review by assigning a rating below 2.5 as negative and a rating above 2.5 as positive.

### Sentiment140

Sentiment140 (Go, Bhayani, and Huang 2009) contains 1.6 million positive and negative tweets, collected and annotated by querying positive and negative emotions, with a tweet considered positive if it contains a positive emoticon like " :) " and negative if, it contains a negative emoticon like " :( ".

### Yelp

We use the dataset collected in (Prusa and Khoshgoftaar 2017). It contains 429,061 Yelp reviews from 12 cities in the United States (Yelp500K). This is an imbalanced dataset with 371,292 positive and 57,769 negative instances. Another 500K reviews were scraped to create a second dataset with a million reviews (Yelp1M).

## General Methodology

We adopt the general convolutional network architecture demonstrated in recent years to advance the state of the art in supervised classification (Krizhevsky and Hinton 2009). We evaluate classification performance with a series of convolutional layers for feature extraction that are followed by two fully-connected layers for classification.

| **Layer** | **Sent140<br>Amazon<br>Yelp** |
|---|---|
| **Input** | 8x140x1<br>8x500x1<br>8x500x1 |
| **Convolutional** | f=64 k=[3,3] |
| **Convolutional** | f=64 k=[3,3] |
| **Convolutional** | f=64 k=[3,3] |
| **Pool** | k=[2,2] s=[2,2] |
| **Convolutional** | f=64 k=[3,3] |
| **Convolutional** | f=64 k=[3,3] |
| **Convolutional** | f=64 k=[3,3] |
| **Pool** | k=[2,2] s=[2,2] |
| **Fully Connected** | n=512 |
| **Drop Out** | kp=0.5 |
| **Fully Connected** | n=2 |

Table 1: ConvNet Configuration. Convolutional layers indicate the number of filters (f=) and the kernel size (k=). Max-pool layers indicate the kernel size (k=) and the stride (s=). Dropout layers show the applied keep probability (kp=), and the fully-connected layers display the number of neurons (n=)

Max-pooling is used between convolutional layers to reduce the dimensionality of the network input, and dropout is used before fully-connected layers to prevent overfitting. Our architecture is presented in Table 1.

Within each dataset, experiments are carried out using the presented CNN architecture, modified only to fit the memory specifications of the activation functions. The only layer that is not modified according to the activation function is the final classification layer, where a softmax activation is applied. The reported results are generated with the models trained using a learning rate of 0.01. Rather than tune each network in our comparison optimally with a validation set, we implement a set of uniform stopping criteria during training to maintain a consistent protocol so that network performance on a test set is suitable for comparison across activations as explained in (Krizhevsky and Hinton 2009). Early stopping criteria is the same for every dataset, and the slope of the test loss is calculated over a running window of the past four epochs. When the slope goes positive, testing loss is no longer decreasing, and network training is stopped. The optimizer is stochastic gradient descent and the loss function is the categorical cross-entropy. Table 2 displays the batch size and number of trainable parameters per dataset.

| Dataset | Batch Size | Number Trainable Parameters |
|---|---|---|
| **Sentiment140** | 200 | 2,743,234 |
| **Amazon** | 200 | 8,641,474 |
| **Yelp** | 200 | 8,641,474 |

Table 2: Batch size and number of trainable parameters per dataset

The classification tasks and datasets used to evaluate the activation functions are two Amazon product data subsets (1 and 4 million reviews), Sentiment140 and two subsets from the Yelp text datasets (500,000 and 1,000,000 reviews). For our sentiment analysis, the text was embedded as proposed by (Prusa and Khoshgoftaar 2016).

ReLU is also evaluated with 2x the number of filters in each convolutional layer. The purpose of including this variant is to consider the impact of increased neurons on the accuracy, training time and memory usage of neural networks independent of the maxout activation. Because maxout incorporates both the max operation and the use of duplicate neurons with additional memory, it is necessary to consider how each component of the activation contributes to its performance.

Maxout is evaluated with the following input feature map-output elements combinations: 2-1, 3-1, 3-2 and 6-1. We compute maxout for our four activations using the equations below, which are suitable for parallelization with modern deep learning software and parallel computer hardware. In general, we use maximum (max) and minimum (min) operations with two inputs to achieve maximum computational efficiency during training.

$$maxout\ 2-1\ (x_1, x_2) = \max(x_1, x_2) \quad (7)$$

$$maxout\ 3-1\ (x_1, x_2, x_3) = \max(x_1, \max(x_2, x_3)) \quad (8)$$

$$maxout\ 6-1\ (x_1, x_2, x_3, x_4 x_5, x_6) = \max(x_1, \max(x_2, \max(x_3, \max(x_4, \max(x_5, x_6))))) \quad (9)$$

$$maxout\ 3-2\ (x_1, x_2, x_3) = \max(x_1, \max(x_2, x_3)), \min(\max(x_1, x_2), \min(\max(x_2, x_3), \max(x_1, x_3))) \quad (10)$$

While it would be ideal to record the wall clock time needed to train each network, modern high-performance computing environments present hardware and software challenges which make it difficult to safely compare training time across runs or activations. Thus, we produce a metric which represents the time cost of training with a particular activation function. This metric is produced for each activation on an isolated desktop computing environment. We record the wall clock time required to train each network in our comparison for 100 batches and average that time across 10 runs on a single desktop computer with 32GB of RAM running Ubuntu 16.04 with an intel i7 7th generation CPU and an NVIDIA 1080ti GPU. Those times are produced independently for each activation on each dataset and are presented in this study as "Avg 100 batches time".

A total of 11 activation functions were evaluated, where each experiment compared:

- Classification accuracy
- Training time (average 100 batches time multiplied by number of epochs to converge)

The training did not converge most of the time with the SeLU activation function on the Amazon and Yelp datasets; out of many runs it was only possible to get one successful training on Amazon1M and Yelp1M. Similarly, maxout 6-1 and maxout 3-2 on Amazon4M failed to converge on large text datasets. Only one run was successful using those activations. Table 3 displays the number of experiments per activation function and dataset.

| Dataset | LR | M21 | M31 | M32 | M61 | R | R2X | SL | T |
|---|---|---|---|---|---|---|---|---|---|
| **Amazon1M** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 |
| **Amazon4M** | 2 | 2 | 2 | 1 | 1 | 2 | 2 | 0 | 2 |
| **Sent140** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| **Yelp500K** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 0 | 2 |
| **Yelp1M** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 |

Table 3: Number of experiments per activation and dataset LReLU (LR), Maxout 2-1 (M21), Maxout 3-1 (M31), Maxout 3-2 (M32), Maxout 6-1 (M61), ReLU (R), ReLU2x (R2X), SeLU (SL), Tanh (T)

In each dataset, we use a train/test split of 90%/10%. Because we apply a consistent early stopping criteria, we report results of our comparison done directly on a test set, without an additional validation set. We implemented our tests in PyTorch (Collobert, Kavukcuoglu, and Farabet 2011).

## Experimental Results

Based on preliminary observations, it was evident the sigmoid activation does not perform well in deep CNNs. For this reason, both activation functions are not presented in the evaluation results.

One-way analysis of variance (ANOVA) is performed to statistically examine the various effects on performances of the type of activation (maxout vs. traditional activation functions) across all the datasets. In this ANOVA test, the results from 82 evaluations were considered together, and all tests of statistical significance utilized a significance level α of 5%. The factor is significant if the *p*-value is less than 0.05. The ANOVA table is presented in Table 4, indicating the activation type does not make a difference in the classification accuracy.

| Factors | Sum of Squares | Percentage of Variation | Degrees of Freedom | Mean Square | F-Computed | *p*-value |
|---|---|---|---|---|---|---|
| Activation Type | 16.56 | 1.04 % | 1 | 16.55 | 0.85 | 0.36 |
| Error | 1566.34 | 98.95 % | 80 | 19.57 | | |
| Total | 1583.9 | 100 % | 81 | | | |

Table 4: One-way ANOVA for type of activation function

The best activation accuracy per dataset, its average 100 batches and training time are presented in Table 5. The training time is number of epochs to converge multiplied by the average time in seconds needed to train 100 batches. ReLU with 2x filters reported the highest accuracy on Sentiment140 and Yelp500K datasets. On Yelp1M, ReLU achieved the highest accuracy and maxout 3-2 and 6-1 reported the best accuracies on the Amazon1M and Amazon4M datasets respectively.

| Dataset | Best Activation | Accuracy | Epochs | Avg 100 Batches Time (s) | Avg 100 Batches Training Time (s) |
|---|---|---|---|---|---|
| Amazon1M | Maxout 3-2 | 88.17 % | 35 | 73.27 | 2124.86 |
| Amazon4M | Maxout 6-1 | 93.73 % | 26 | 57.32 | 1490.36 |
| Sentiment140 | ReLU2x | 84.57 % | 60 | 5.09 | 259.79 |
| Yelp500K | ReLU2x | 93.17 % | 60 | 18.19 | 873.33 |
| Yelp1M | ReLU | 93.60 % | 60 | 8.65 | 519.41 |
| All datasets combined | ReLU2x | 90.41 % | 40 | 15.57 | 594.15 |

Table 5: Best activation accuracy, average 100 batches and training time per dataset

On average, ReLU2x reported the highest accuracy of 90.41%. SeLU was difficult to train, with convergence of the deep network occurring only for the Sentiment140 dataset. Adding a layer of filters was enough for ReLU to achieve the higher classification accuracy. This suggests that adding more layers could increase the accuracy but hyperparameter tuning might be required.

We performed Tukey's Honestly Significant Difference (HSD) test to further investigate these results. The HSD is a statistical test comparing the mean value of the performance measure for the different activation functions. All tests of statistical significance use a significance level α of 5%. Two activation functions with the same block letter are not significantly different with 95% statistical confidence (e.g. group a is significantly different than group b). In Table 6, the letters in the third column indicate the HSD grouping of the activation accuracy. That is, if two activations have the same letter in the HSD column, their accuracies are not significantly different. The HSD test shows six activations are statistically indistinguishable from one another (they all have the block letter 'a' in the HSD column). ReLU2x, with the exception of the Yelp1M dataset, recorded the highest or second highest accuracy, while maxout 3-2 recorded within the top three accuracies except on the Sentiment140 and Yelp1M datasets. All maxout and ReLU activations are not significantly different from each other but they are statistically different from tanh and SeLU.

Maxout activations had a higher average 100 batches training time than ReLU and ReLU2x. The lowest average 100 batches time was ReLU with 7.412 seconds, which surprisingly delivered the third highest average classification accuracy. This suggests that a higher training time will not always deliver better results.

| Activation | Accuracy | Accuracy HSD | Avg 100 Batches Time (s) | Avg 100 Batches Training Time (s) |
|---|---|---|---|---|
| ReLU 2x | 90.41 % | a | 15.57 | 594.15 |
| Maxout 3-2 | 90.35 % | a | 62.67 | 1485.35 |
| ReLU | 90.26 % | ab | 7.41 | 349.52 |
| Maxout 3-1 | 90.19 % | ab | 29.67 | 972.79 |
| Maxout 2-1 | 89.97 % | ab | 17.50 | 440.28 |
| Maxout 6-1 | 89.89 % | ab | 48.83 | 1866.42 |
| LReLU | 89.71 % | b | 13.86 | 754.60 |
| Tanh | 87.57 % | c | 7.45 | 242.19 |
| SeLU | 83.81 % | d | 12.20 | 229.62 |

Table 6: Activation HSD test, 100 batches average and training time

## Conclusion

We conducted experiments to assess the effectiveness of maxout variants. Doubling the number of convolutional filters on ReLU on big data is very conclusive. It suggests that given a sentiment analysis task and a CNN architecture, ReLU2x and maxout 3-2 are likely to produce the highest classification accuracy results compared to low memory usage activation functions and the rest of the maxout variants analyzed in this study. It is important to note that the difference between ReLU and ReLU2x is the choice of a tunable hyperparameter. In this study due to hardware memory constraints, we exclude ReLU3x and ReLU6x from comparisons.

Maxout variants provided a better average accuracy than LReLU, SeLU and tanh as shown in Table 6, but on average the training time is slower except for maxout 2-1 compared to LReLU. Maxout 3-2 was the only variant that outperformed the simple Max-Feature-Map. Results indicate that ReLU, with more filters, was the top performer with the tradeoff of high memory usage and big data that is difficult to train. A maxout variant performed better than the rest of the activation functions with the Amazon datasets. Although ReLU2x provided the best average accuracy, the maxout variants were very close to ReLU2x in performance. On average, ReLU2x tends to converge 2.5x faster than maxout 3-2 but it is 1.7x slower than ReLU. There is no relationship between the activation functions that use more memory or have a higher training time and the classification accuracy performance, but clearly adding more convolutional filters enhanced ReLU, which did not happen with maxout. ReLU is the recommended activation function due to high performance, with both tested number of filters (ReLU and ReLU2x), and fast training relative to other top performing activations.

Future work will involve conducting additional empirical studies with ReLU3x and ReLU6x on big data and hyperparameter tuning recommendations that were outside the scope of this work. Also, future work could include additional deep network architectures and domains.

## References

Baziotis, C.; Pelekis, N.; and Doulkeridis, C. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017).*

Cai, M.; Shi, Y.; and Liu, J. 2013. Deep maxout neural networks for speech recognition. In *Proceedings ASRU.*

Collobert, R.; Kavukcuoglu, K.; and Farabet, C. 2011. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS workshop.*

Go, A.; Bhayani, R.; and Huang, L. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford* 1(12).

Goodfellow, I.; Mirza, M.; Xiao, D.; Courville, A.; and Bengio, Y. 2013. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211.*

Goodfellow, I.; Warde-Farley, D.; Mirza, M.; Courville, A.; and Bengio, Y. 2013. Maxout Networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013).*

Heredia, B.; Khoshgoftaar, T.; Prusa, J.; and Crawford, M. 2016. Integrating multiple data sources to enhance sentiment prediction. In *Collaboration and Internet Computing (CIC), 2016 IEEE 2nd International Conference.*

Jebbara, S., and Cimiano, P. 2017. Aspect-Based Relational Sentiment Analysis Using a Stacked Neural Network Architecture. *arXiv preprint arXiv:1709.06309.*

Johnson, R., and Zhang, T. 2014. Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058.*

Klambauer, G.; Unterthiner, T.; Mayr, A.; and Hochreiter, S. 2017. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, 971-980.

Krizhevsky, A., and Hinton, G. 2009. Learning multiple layers of features from tiny images. *Technical report, University of Toronto* 1(4): 7.

LeCun, Y., and Bengio, Y. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks* 3361(10).

Liew, S.; Khalil-Hani, M.; and Bakhteri, R. 2016. Bounded activation functions for enhanced training stability of deep neural networks on visual pattern recognition problems. *Neurocomputing* 216:718-734.

Maas, A.; Hannun, A.; and Ng, A. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings ICML.*

McAuley, J.; Pandey, R.; and Leskovec, J. 2015. Inferring networks of substitutable and complementary products. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'15).*

Nair, V., and Hinton, G. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10).*

Njikam, A., and Zhao, H. 2016. A novel activation function for multilayer feed-forward neural networks. *Applied Intelligence* 45(1): 75-82.

Poria, S.; Cambria, E.; and Gelbukh, A. 2015. Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis. In *Conference on empirical methods in natural language processing.*

Prusa, J. D., and Khoshgoftaar, T. M. 2017. Training Convolutional Networks on Truncated Text. In *Proceedings of the IEEE International Conference on Tools with Artificial Intelligence.*

Prusa, J. D., and Khoshgoftaar, T. M. 2016. Designing a Better Data Representation for Deep Neural Networks and Text Classification. In *Information Reuse and Integration (IRI), 2016 IEEE 17th International Conference.*

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15(1): 1929-1958.

Wu, X.; He, R.; Sun, Z.; and Tan, T. 2015. A light CNN for deep face representation with noisy labels. *arXiv preprint arXiv:1511.02683.*

Zhang, X.; Trmal, J.; Povey, D.; and Khudanpur, S. 2014. Improving deep neural network acoustic models using generalized maxout networks. In *IEEE International Conference* In *Acoustics, Speech and Signal Processing (ICASSP).*