Effectiveness of Self Normalizing Neural Networks for Text Classification

Avinash Madasu and Vijjini Anvesh Rao

Samsung R&D Institute, Bangalore
{m.avinash,a.vijjini}@samsung.com

Abstract. Self Normalizing Neural Networks(SNN) proposed on Feed Forward Neural Networks(FNN) outperform regular FNN architectures in various machine learning tasks. Particularly in the domain of Computer Vision, the activation function Scaled Exponential Linear Units (SELU) proposed for SNNs, perform better than other non linear activations such as ReLU. The goal of SNN is to produce a normalized output for a normalized input. Established neural network architectures like feed forward networks and Convolutional Neural Networks(CNN) lack the intrinsic nature of normalizing outputs. Hence, requiring additional layers such as Batch Normalization. Despite the success of SNNs, their characteristic features on other network architectures like CNN haven't been explored, especially in the domain of Natural Language Processing. In this paper we aim to show the effectiveness of proposed, Self Normalizing Convolutional Neural Networks(SCNN) on text classification. We analyze their performance with the standard CNN architecture used on several text classification datasets. Our experiments demonstrate that SCNN achieves comparable results to standard CNN model with significantly fewer parameters. Furthermore it also outperforms CNN with equal number of parameters.

1 Introduction

The aim of Natural Language Processing(NLP) is to analyze and extract information from textual data in order to make computers understand language, the way humans do. Unlike images which lack sequential patterns, texts involve amplitude of such information which makes processing very distinctive.
The level of processing varies from paragraph level, sentence level, word level and to the character level. Deep neural network architectures achieved state-of-art results in many areas like Speech Recognition [1] and Computer Vison [2]. The use of neural networks in Natural language processing can be traced back to [3] where the backpropogation algorithm was used to make networks learn familial relations. The major advancement was when [4] applied neural networks to represent words in a distributed compositional manner. [5] proposed two neural network models CBoW and Skip-gram for an efficient distributed representation of words. This was a major break-through in the field of NLP. From then, neural network architectures achieved state-of-results in many NLP applications like Machine Translation [6], Text Summarization [7] and Conversation Models [8] .

Convolutional Neural Networks [9] were devised primarily for dealing with images and have shown remarkable results in the field of computer vision[10, 11]. In addition to their contribution in Image processing, their effectiveness in Natural language processing has also been explored and shown to have strong performance in Sentence[12] and Text Classification[13].

The intuition behind Self Normalizing Neural Networks(SNN) is to drive neuron activations across all layers to emit a zero mean and unit variance output. This is done with the help of the proposed activation in SNNs, SELU or scaled exponential linear units. With the help of SELUs an effect alike to batch normalization is replicated, hence slashing the number of parameters along with a robust learning. Special Dropouts and Initialization also help in this learning, which make SNNs remarkable to traditional Neural Networks. As Image based inputs and Text based inputs differ from each other in form and characteristics, in this paper we propose certain revisions to the SNN architecture to empower them on texts efficiently.

In this paper, to explore effectiveness of self normalizing neural networks in text classification, we propose an architecture, Self Normalizing Convolutional Neural Network (SCNN) built upon convolutional neural networks. A thorough study of SCNNs on various benchmark text datasets, is paramount to ascertain importance of SNNs in Natural Language Processing.

2 Related Work

Prior to the success of deep learning, text classification heavily relied on good feature engineering and various machine learning algorithms.

Convolutional Neural Networks [9] were devised primarily for dealing with images and have shown remarkable results in the field of computer vision[10, 11]. In addition to their contribution in Image processing, their effectiveness in Natural language processing has also been explored and shown to have strong performance. Kim [12] represented an input sentence using word embeddings that are stacked into a two dimensional structure where length corresponds to embedding size and height with average sentence length. Processing this structure using kernel filters of fixed window size and max pooling layer upon it to capture the most important information has shown them promising results on text classification. Additionally, very deep CNN architectures [13] have shown state-of-the art results in text classification, significantly reducing the error percentage. As CNNs are limited to fixed window sizes, [14] have proposed a recurrent convolution architecture to exploit the advantages of recurrent structures that capture distant contextual information in ways fixed windows may not be able to.

Klambauer [15] proposed Self Normalizing Neural Networks (SNN) upon feed forward neural networks, significantly outperformed FNN architectures on various machine learning tasks. Since then, the activation proposed in SNNs, SELU have been widely studied in Image Processing[16–18], where they have been applied on CNNs to achieve better results. SELU's effectiveness have also been

explored in Text[19–21] processing tasks. However these applications are limited to applying just SELUs in their [16–21] respective architectures.

3 Self-Normalizing Neural Networks

Self-Normalizing Neural Networks(SNN) are introduced by Gnter Klambauer [15] to learn higher level abstractions. Regular neural network architectures like Feed forward Neural Networks(FNN), Convolutional Neural Networks(CNN) lack the property of normalizing outputs and require additional layers like Batch Normalization[22] for normalizing hidden layer outputs. SNN are specialized neural networks in which the neuron activations automatically converge to a fixed mean and variance. Training of deep CNNs can be efficiently stabilized by using batch normalization and by using Dropouts [23]. However FNN suffer from high variance when trained with these normalization techniques. In contrast, SNN are very robust to high variance thereby inducing variance stabilization and overcoming problems like exploding gradients[15]. SNN differs from naive FNN and CNN by the following:

3.1 Input Normalization

To get a normalized output in SNN without requiring layers like batch normalization, the inputs are normalized.

3.2 Intialization

Weights initialization is an important step in training neural networks.Several initialization methods like glorot uniform[24] and lecun normal[25] have been proposed. FNN and CNN are generally initialized using glorot uniform whereas SNN are initialized using lecun normal. Glorot uniform initialization draws samples centered around 0 and with standard deviation as:

$$stddev = \sqrt{\frac{2}{(in + out)}} \tag{1}$$

Lecun normal initialization draws samples centered around 0 and with standard deviation as:

$$stddev = \sqrt{\frac{1}{in}} \tag{2}$$

where *in* and *out* represent dimensions of weight matrix corresponding to number of nodes in previous and current layer respectively.

3.3 SELU activations

Scaled exponential linear units (SELU) is the activation function proposed in SNNs. In general, FNN and CNN use rectified linear units(ReLU) as activation. ReLU activation clips negative values to 0 and hence suffers from dying ReLU problem[1]. As explained by [15] an activation function should contain both positive and negative values for controlling mean, saturation regions for reducing high variance and slope greater than one to increase variance if its value is too small. Hence SELU activation is introduced to preserve the aforementioned properties. SELU activation function is defined as :

$$selu(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha e^x - \alpha & \text{if } x \leqslant 0 \end{cases} \tag{3}$$

where x denotes input, α ($\alpha = 1.6733$), λ ($\lambda = 1.0507$) are hyper parameters, and e stands for exponent

3.4 Alpha Dropout

Standard dropout[23] drops neurons randomly by setting their weights to 0 with a probability $1 - p$. This prevents the network to set mean and variance to a desired value. Standard dropout works very well with ReLUs because in them, zero falls under the low variance region and is the default value. For SELU, standard dropout does not fit well because the default low variance is $\lim_{x \to \infty} selu(x) = - \lambda\alpha = \alpha^{'}$[15]. Hence alpha dropout is proposed which sets input values randomly to $\alpha^{'}$. Alpha dropout restores the original values of mean and variance thereby preserving the self-normalizing property [15]. Hence, alpha dropout suits SELU by making activations into negative saturation values at random.

4 Model

We propose Self-normalizing Convolutional Neural Network (SCNN) for text classification as shown in the figure 1. To show the effectiveness of our proposed model, we adapted the standard CNN architecture used for text classification to SCNN with the following changes:

4.1 Word Embeddings are not normalized

Self-Normalizing Neural Networks require inputs to be normalized for the outputs to be normalized[15]. Normalization of inputs work very well in computer vision because images are represented as pixel values which are independent of the neighbourhood pixels. In contrast, word embedding for a particular word is

[1] http: //cs231n.github.io/neural-networks-1/

created based on its co-occurrence with context. Words exhibit strong dependency with their neighbourhood words. Similar words contain similar context and are hence close to each other in their embedding space. If word embeddings are normalized, the dependencies are disturbed and their normalized values will not represent the semantics behind the word correctly.

4.2 ELU activation as an alternative to SELU

SELU activation originally proposed for SNN[15] preserve the properties of SNN if the inputs are normalized. When inputs are normalized, applying SELU on the activations does not shift the mean. However if inputs weren't normalized, due to the parameter λ in SELU activation, the neuron outputs will be scaled by a factor λ thereby shifting the mean and variance to a value away from the desired mean and variance. These values are further propagated to other layers thereby shifting mean and variance more and more. Since input word embeddings cannot be normalized as explained in section 4.1, we use ELU activation [26] in the proposed SCNN model instead. ELU activation function is defined as :

$$elu(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha e^x - \alpha & \text{if } x \leqslant 0 \end{cases} \tag{4}$$

where α is a hyper parameter, and e stands for exponent. The absence of parameter λ in ELU prevents greater scaling of neuron outputs. ELU activation pushes the mean of the activations closer to zero even if inputs are not normalized which enable faster learning [26]. We compare the performance of SCNN with both SELU and ELU activations and the results presented in table 3 and figure 2.

4.3 Model Architecture

The SCNN architecture is shown in the figure 1. Let V be the vocabulary size considered for each dataset and $X \in \mathbb{R}^{V \times d}$ represent the word embedding matrix where each word X_i is a d dimensional word vector. Words present in the pretrained word embedding[2] are assigned their corresponding word vectors. Word that are not present are initialized to 0s. Based on our experiments, SCNN showed better performance when absent words are initialized to 0s than randomly initialization. A maximum sentence length of N is considered per sentence or paragraph. If the sentence or paragraph length is less than N, zero padding is done. Therefore, $I \in \mathbb{R}^{N \times d}$ dimensional vector per each sentence or paragraph is provided as input to the SCNN model.

Convolution operation is applied on I with kernel $K \in \mathbb{R}^{h \times d}$(h $\in$ {3,4,5}) is applied to input vectors with a window size of h. The weight initialization of these kernels is done using lecun normal [25] and bias is initialized to 0. A new

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

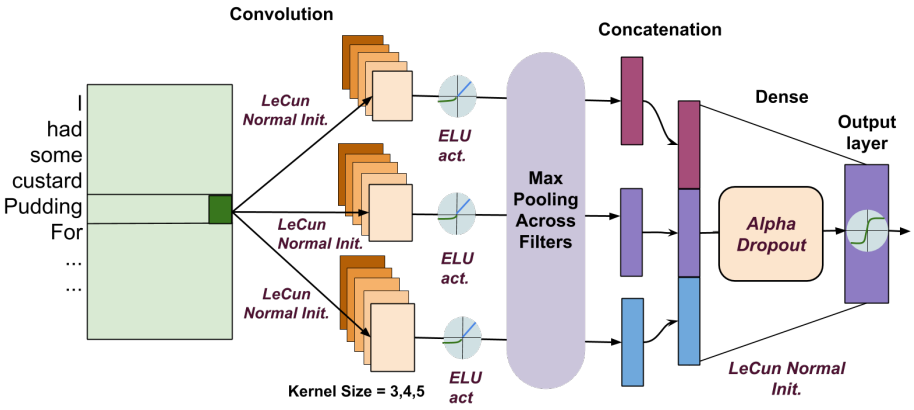


Fig. 1: Architecture of proposed SCNN model

feature vector is $C \in \mathbb{R}^{(N-h+1)\times 1}$ is obtained after the convolution operation for each filter.

$$C = f(I \circledast K) \tag{5}$$

where f represents the activation function (ELU). Number of convolution filters vary depending on the dataset, table 2 summarizes the number of parameters for all of our experiments. Maxpooling operation is applied across each filter C to get the maximum value. The outputs from the maxpooling layer across all filters are concatenated. Alpha dropout [15] with a dropout value 0.5 is applied on the concatenated layer. The concatenated layer is densely connected to the output layer with activation as sigmoid if task is binary classification and softmax otherwise.

5 Experiments and Datasets

Table 1: Summary statistics of all datasets

Datasets	No. of classes	Dataset Size	Test
MR	2	10662	10 fold Cross Validation
SO	2	10000	10 fold Cross Validation
IMDB	2	50000	25000
TREC	6	5952	500
CR	2	3773	10 fold Cross Validation
MPQA	2	10604	10 fold Cross Validation

Table 2: Parameters of examined models on all datasets

Datasets	**No of Conv.Filters**		**No of Parameters**	
	SCNN and Short-CNN	Static CNN[12]	SCNN and Short-CNN	Static CNN[12]
MR	210	300	$\approx 254k$	$\approx 362k$
SO	210	300	$\approx 254k$	$\approx 362k$
IMDB	210	300	$\approx 254k$	$\approx 362k$
TREC	210	300	$\approx 254k$	$\approx 362k$
CR	90	300	$\approx 108k$	$\approx 362k$
MPQA	90	300	$\approx 108k$	$\approx 362k$

5.1 Datasets

We performed experiments on various benchmark data sets of text classification. The summary statistics for the datasets are shown in table 1

Movie Reviews(MR) It consists of 10662 movie reviews with 5331 positive and 5331 negative reviews[27]. Task involves classifying reviews into positive or negative sentiment.

Subjectivity Objectivity(SO) The dataset consists of 10000 sentences with 5000 subjective sentences and 5000 objective [28]. It is a binary classification task of classifying sentences as subjective or objective.

IMDB Movie Reviews(IMDB) The dataset consists of 50000 movie reviews of which 25000 are positive and 25000 are negative [29].

TREC TREC dataset contains questions of 6 categories based on the type of question: 95 questions for Abbreviation,1344 questions for Entity, 1300 questions for Description, 1288 questions for Human, 916 questions for Location and 1009 questions for Numeric Value [30].

Customer Reviews(CR) The dataset consists of 2406 positive reviews and 1367 negative reviews. It is a binary classification task of predicting positive or negative sentiment [31].

MPAQ The dataset consists of 3311 positive reviews and 7293 negative reviews. Binary classification task of predicting positive and negative opinion [32].

5.2 Baseline Models

We compare our proposed model SCNN with the following models:

Static CNN model We compare SCNN with the static model, one of the standard CNN models proposed for text classification [12].

SCNN with SELU activation SNN was originally proposed with SELU activation. We performed experiments on SCNN using SELU as the activation function in place of ELU.

Short CNN Our model SCNN is proposed with fewer parameters compared to Static CNN model[12]. To show the effectiveness of SCNN, we perform experiments on Static CNN model with same number of parameters as SCNN, we refer this model as Short CNN.

5.3 Model parameters

Table 2 shows the parameter statistics for all the models. SCNN and Short CNN models are experimented using same number of parameters. We used 70 convolution filters for each kernel in case of MR, SO, IMDB and TREC datasets. For datasets CR and MPQA we considered 30 filters for each kernel. In MPQA dataset, the average sentence length is 3. Hence, we reduced the convolution filters from 70 to 30.

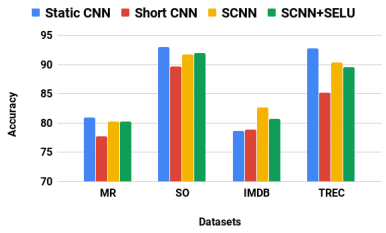
5.4 Training

We process the dataset as follows: Each sentence or paragraph is converted to lower case. Stop words are not removed from the sentences. We consider a vocabulary size V for each dataset based on the word counts. The datasets IMDB, TREC have predefined test data and for other datasets we used 10-fold Cross Validation. The parameters chosen vary depending on the dataset size. Table 2 shows the parameters of SCNN model for all datasets. We used Adam [33] as the optimizer for training SCNN.

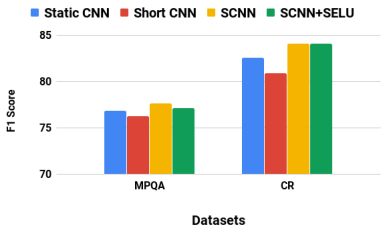
6 Results and Discussion

6.1 Results

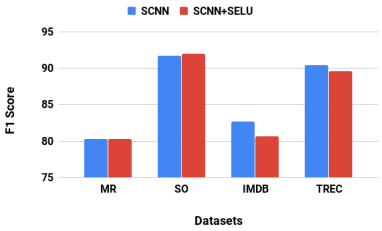
The performance of the models for all datasets is shown in table 3. For all the balanced datasets MR, SO, IMDB and TREC, accuracy is used as the metric for comparison. The performance comparison on imbalanced datasets like CR and MPQA cannot be justified using accuracy because imbalance can induce bias in the models' prediction. Hence we use F1-Score as the metric for performance analysis for CR and MPQA. The datasets IMDB and TREC have preexisting train, test sets. Therefore, we report our results on the provided test sets for them. For remaining datasets, we report results using 10-fold cross validation(CV).



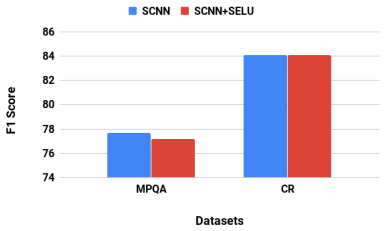
(a) Accuracy Score comparison on MR,SO,IMDB,TREC Datasets



(b) F1-Score comparison on CR, MPQA Datasets



(c) Performance comparison of SELU and ELU on MR,SO,IMDB,TREC Datasets



(d) Performance comparison of SELU and ELU on CR,MPQA Datasets Length

Fig. 2: Figures to demonstrate performance of all models

6.2 Discussion

SCNN models against Short-CNN:
When we compare SCNN with SELU and SCNN to Short CNN, both the models of SCNN outperform Short CNN for all the datasets. This shows that SCNN models perform better than CNN models (Short CNN) with same number of parameters indicating a better generalization of training. There is a significant improvement in accuracy and F1-Score when SCNN models are used in place of CNN. We believe that the use of activation functions ELU and SELU in the SCNN models as opposed to ReLU is the leading factor behind this performance difference between SCNN and CNN. In particular, ReLU activation suffers from dying ReLU problem[3]. In ReLU, the negative values are cancelled to 0. Therefore negative values in the pretrained word vectors are ignored thereby loosing information about negative values. This problem is solved in ELU and SELU by having activation even for the negative values. In comparison to ReLU, ELU and SELU have faster and accurate training convergence leading to better generalization performance.

ELU against SELU in SCNN:
We proposed SCNN using ELU as activation function opposed to SELU, the

[3] http: //cs231n.github.io/neural-networks-1/

Table 3: Performance of the models on different datasets

Model	**Datasets**					
	MR	SO	IMDB	TREC	CR	MPQA
	Accuracy				F1-Score	
Short CNN	77.762	89.63	78.84	85.2	76.246	80.906
SCNN w/SELU	80.266	**91.99**	80.664	89.6	77.166	84.062
SCNN	**80.308**	91.759	**82.708**	**90.4**	**77.666**	**84.068**
CNN-static[12]	81	93	78.692	92.8	76.852	82.584

activation function introduced originally for SNN. We found that if ELU is used as activation, the performance of SCNN is better for a majority of the datasets. Our results from table 3 substantiate the claim about the effectiveness of ELU activation. The characteristic difference between SELU and ELU activations is the parameter λ ($\lambda > 1$) in SELU activation which scales the neuron outputs. SELU is effective for maintaining normalized mean and variance when the inputs are normalized. Since, pretrained word vectors are not normalized, the parameter λ adversely scales the outputs. This results in a shifted mean and variance from the desired values. On propagation through subsequent layers the difference only gets further magnified. On the other hand, ELU pushes the activations to unit mean even if the inputs are not normalized. Hence, ELU achieved better results compared to SELU activation in SCNN.

SCNN against static CNN:
Our results from table 3 indicate that SCNN achieves comparable results to Static CNN. As shown in table 2, the stark difference in the parameter counts between SCNN and static CNN is more than a million. For datasets IMDB, CR and MPQA, SCNN outperforms static CNN. In case of MR dataset, performance difference between SCNN and static CNN is very minimal.

7 Conclusion

We propose SCNN for performing text classification. Our observations indicate that SCNN has comparable performance to CNN (Static-CNN [12]) model with substantially lesser parameters. Moreover SCNN performs significantly better than CNN with equal number of parameters. The experimental results demonstrate the effectiveness of self normalizing neural networks in text classification. Currently, SCNN is proposed with relatively simple architectures. Our work can be further extended by experimenting SCNN on deep architectures. In addition to this, SNN can also be applied on recurrent neural networks(RNN) and its performance can be analyzed.

References

1. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal processing magazine **29** (2012) 82–97
2. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
3. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. nature **323** (1986) 533
4. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. Journal of machine learning research **3** (2003) 1137–1155
5. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
6. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
7. Rush, A.M., Chopra, S., Weston, J.: A neural attention model for abstractive sentence summarization. arXiv preprint arXiv:1509.00685 (2015)
8. Vinyals, O., Le, Q.: A neural conversational model. arXiv preprint arXiv:1506.05869 (2015)
9. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86** (1998) 2278–2324
10. Krizhevsky, A., Sutskever, I., E. Hinton, G.: Imagenet classification with deep convolutional neural networks. Neural Information Processing Systems **25** (2012)
11. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
12. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
13. Conneau, A., Schwenk, H., Barrault, L., Lecun, Y.: Very deep convolutional networks for text classification. arXiv preprint arXiv:1606.01781 (2016)
14. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent convolutional neural networks for text classification. In: AAAI. Volume 333. (2015) 2267–2273
15. Klambauer, G., Unterthiner, T., Mayr, A., Hochreiter, S.: Self-normalizing neural networks. In Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., eds.: Advances in Neural Information Processing Systems 30. Curran Associates, Inc. (2017) 971–980
16. Lguensat, R., Sun, M., Fablet, R., Tandeo, P., Mason, E., Chen, G.: Eddynet: A deep neural network for pixel-wise classification of oceanic eddies. In: IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, IEEE (2018) 1764–1767
17. Zhang, J., Shi, Z.: Deformable deep convolutional generative adversarial network in microwave based hand gesture recognition system. In: Wireless Communications and Signal Processing (WCSP), 2017 9th International Conference on, IEEE (2017) 1–6
18. Goh, G.B., Hodas, N.O., Siegel, C., Vishnu, A.: Smiles2vec: An interpretable general-purpose deep neural network for predicting chemical properties. arXiv preprint arXiv:1712.02034 (2017)

19. Kumar, S.S., Kumar, M.A., Soman, K.: Sentiment analysis of tweets in malayalam using long short-term memory units and convolutional neural nets. In: International Conference on Mining Intelligence and Knowledge Exploration, Springer (2017) 320–334
20. Rosá, A., Chiruzzo, L., Etcheverry, M., Castro, S.: Retuyt in tass 2017: Sentiment analysis for spanish tweets using svm and cnn. arXiv preprint arXiv:1710.06393 (2017)
21. Meisheri, H., Ranjan, K., Dey, L.: Sentiment extraction from consumer-generated noisy short texts. In: Data Mining Workshops (ICDMW), 2017 IEEE International Conference on, IEEE (2017) 399–406
22. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. (2015) 448–456
23. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15** (2014) 1929–1958
24. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In Teh, Y.W., Titterington, M., eds.: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. Volume 9 of Proceedings of Machine Learning Research., Chia Laguna Resort, Sardinia, Italy, PMLR (2010) 249–256
25. LeCun, Y., Bottou, L., Orr, G.B., Müller, K.R.: Efficient backprop. In: Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop, London, UK, UK, Springer-Verlag (1998) 9–50
26. Clevert, D., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). CoRR **abs/1511.07289** (2015)
27. Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the ACL. (2005)
28. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the ACL. (2004)
29. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA, Association for Computational Linguistics (2011) 142–150
30. Li, X., Roth, D.: Learning question classifiers. In: Proceedings of the 19th International Conference on Computational Linguistics - Volume 1. COLING '02, Stroudsburg, PA, USA, Association for Computational Linguistics (2002) 1–7
31. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2004) 168–177
32. Wiebe, J., Wilson, T., Cardie, C.: Annotating expressions of opinions and emotions in language. Language resources and evaluation **39** (2005) 165–210
33. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014)