International Journal of Engineering and Applied Physics (IJEAP)
Vol. 1, No. 2, May 2021, pp. 96~102
ISSN: 2737-8071

Analysis of User's Opinion using Deep Neural Network Techniques

V. Harini[1], K. Rajalakshmi[2], G.S. Varsha[3]
[1,2,3]Dept. Of Computer Science and Engineering at Thiagarajar College of Engineering, Madurai, India.

Article Info

Article history:

Received Sep 5, 2020
Revised Feb 20, 2021
Accepted Mar 8, 2021

Keywords:

Aspect-specific
Proximity-weighted convolution neural network
Syntax free
Semantic-relatedness

ABSTRACT

Through many research and discoveries it has been widely accepted that aspect-level sentiment classification is achieved effectively by using Long Short-Term Memory (LSTM) network combined with attention mechanism and memory module. As existing approaches widely depend on the modelling of semantic relatedness of an aspect, at the same time we ignore their syntactic dependencies which is already a part of that sentence. This will result in undesirably an aspect on textual words that are descriptive of other aspects. So, in this paper, to offer syntax free contexts as well as they should be aspect specific, so we propose a proximity-weighted convolution network. To be more precise, we have one way of determining proximity weight which is dependency proximity. The construction of the model includes a bidirectional LSTM architecture along with a proximity-weighted convolution neural network.

Corresponding Author:

V.Harini,
Computer Science and Engineering,
Thiagarajar College of Engineering,
Madurai, India.
Email: harinivijayakumara@gmail.com

1. INTRODUCTION

Aspect-based Sentiment Classification is a fine-grained analysis that aims to detect the polarity of a target within a given context that is either a comment (or) review. For example," *The ingredients added to the Italian food are tasty and spicy but it doesn't suit South Indian dishes*". Here the sentiment polarities for aspects '*Italian food*', '*South Indian dishes*' and '*Ingredients*' are positive, negative and neutral respectively. Natural Language Processing (NLP) and Information Retrieval (IR), and plays a vital role in numerous fields like recommendation systems. Earlier works in this area focus on manual extraction of refined features and feed them into classifiers like Support Vector Machine (SVM), which is usually labour intensive. So, to face this emerging problem, extraction of features in automatic ways have been investigated. For example, K.Xu [2] proposed to calculate the sentiment of textual words to the aspect based on their syntactic relationships. ChenZhang et al. [22] built a syntax-aware feature extractor to discover the relevant features. Despite these approaches being effective, A.Abdi et al. [14] argued that the modelling of semantic relatedness of an aspect and its textual words remained a dare, and suggested using a target-dependent LSTM network to label this dare.

Having the ability to model semantic relationships between aspects and their related textual words, these models have gained performance over previous methodologies. Though, they ignore the syntactic connections between the aspect and its text-based words, which will block the adequacy of perspective based literary portrayals. For illustration, a given aspect may show up on various textual words that are shown close to the perspective yet don't relate with the viewpoint grammatically. For example, in "*Its size is absolute and the weight is admissible*", the aspect term *size* may easily be expressed by *admissible* based on semantic relatedness, but that is not the case. Syntactic parsing has been used in earlier approaches, though, the word-level parsing will obstruct feature extraction over a variety of phrases, as the polarity of an aspect is calculated by a key phrase instead of a single word. To achieve the limitations declared above, we propose an aspect-based sentiment classification framework that supports the syntactic relationships between an aspect and its textual words and builds the features at the n-gram level, inside an LSTM-based architecture.

Encouraged by the dependency proximity mechanism, the framework uses a textual word's syntactic proximity to the aspect, also known as proximity weight, to calculate its weightage in the sentence. We then make the proximity weights into a convolution network to know n-gram information, called as Proximity-Weighted Convolution Network (PWCN). Towards the end, a layer of max-pooling is taken on to select the most promising features for prediction.

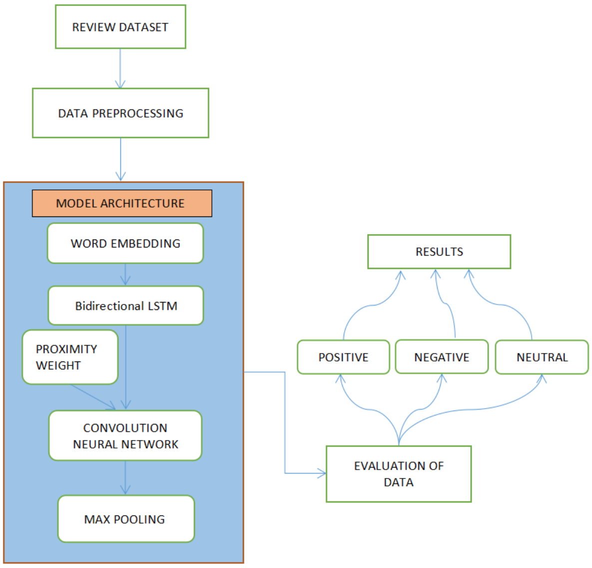


Figure 1. Proposed model architecture.

Demonstrations are regulated on SemEval 2014 Task4 Datasets. The obtained results depict that our model attains a greater performance than a limit of state-art models, and therefore demonstrate that syntactical dependencies are more advantageous than semantic relatedness to aspect-based sentiment classification.

2. ANALYSIS OF SENTIMENTS BASED ON ASPECTS

Analysis of sentiments based on aspects is a sub-task of sentiment analysis which gives a cavernous understanding of the works. For instance, '*The user interface is worse, but the reliability is good.*', for the *user interface* aspect, the polarity is negative whereas *reliability* is positive. To declare a polarity for a particular text, it should be more necessary to understand the text initially by finding out the aspect.

When we start focusing on aspects we will find it easier to calculate the polarity and further determine the sentiments most accurately. This kind of follow up will help people who read reviews and wanted to get to know the end opinion of the overall reviews and to decide whether to acquire the particular product or not. If in case, we try to find the sentiment for the entire statement we will end up in contradictions most of the time. For example, '*The food is yummy but at the same time, the veggies present in it are half baked*', if we calculate the sentiment for the entire statement we will end up with the result as 'neutral' as we have both positive and negative words. But if we concentrate on aspects, here it's '*food*' and '*ingredients*' they result as positive and negative as the word '*yummy*' will have a positive sign concerning food and the word '*half-baked*' will be a negative word for ingredients. So, people will get a clear picture of how to categorize them after they are been classified with the help of our suggested model.

To achieve this, we thought of implementing a convolution neural network (CNN) where we will have to find the proximity weight initially and proceed to classify the user's opinion. The proximity weight calculation is done based on dependency which involves distance vectors. The distance vectors are measured from the main aspect terms for the verbs, adjectives and adverbs. The most important thing is to train the model in such a way that it identifies noun as the aspects and the rest as its fields to which they should be mapped.

So, calculating sentiments is not a big deal but it should be worth as when it is focused on aspects. Moreover, it should not only focus on aspects but also its subordinates so that the distance vector graph that is drawn will be fruitful. At the same time, calculating sentiments for 'n' reviews should be made possible as it will help the end-user know about the product or an article that they are in search of. To accomplish all the above said, we have implemented a super-do algorithm that can overcome all the limitation that was found by other researchers and can remain successful as always.

3. THE PROPOUND MODEL

An outline of our suggested model is given in Figure 1. In this model, an n-word phrase having a target m-word aspect term is composed that denotes the start token of the aspect term. Each word is implanted into a low-dimensional real-valued vector with a matrix. Then word vectors are acquired by word embedding, a bidirectional LSTM is acquired to produce the hidden state vectors. Specifically, $h_i \in R^{2d}$ is a sequence of hidden states respectively acquired from the forward LSTM and the backward LSTM, where d is the dimensionality of a hidden state vector in a unidirectional LSTM. The representation of the hidden state is enhanced by the proximity weighted convolution and used to predict sentiment polarity.

4. OVERVIEW OF PROXIMITY WEIGHT

Earlier attention-based models basically aim at how to acquire a textual depiction based on its integrant words' semantic relation with a corresponding aspect. These models compute attention weights mentioning word vector depiction in the inactive semantic space aside from syntax information. This might restrict the efficiency of these models to misidentify pivotal textual words for identifying the aspect. So, we replace this intricate modelling of aspects by assimilating the syntactical dependencies to exhibit integrant words' characteristics to the aspect. That syntactical dependency information in our model is formulated as proximity weight, which depicts the textual words' proximity to the aspect. Remember the instance related to the weight of a gadget saying that "*Its size is absolute and the weight is admissible*." The bunch of words including {*absolute, admissible*} which are nearer to the aspect term *weight* in respect of semantics, should have a greater probability depicting the weight of a gadget. Further, from the viewpoint of syntax parsing, *absolute* can be securely eliminated from the set of words as it is syntactically too far away from weight, specifying a positive sentiment.

Supporting this idea, we propose a unique method which is dependency proximity, to train the syntactical dependency between textual words and the aspect term respectively.

4.1. DEPENDENCY PROXIMITY

Aside from the decided position in the context, we also contemplate measuring the distances between words in the syntax dependency parsing tree. For instance, in a review "*The food was delicious-must try the mint curry.*" with *food* as the aspect, we initially construct a dependency tree, then calculate for a textual word the tree-based distance, i.e., the length of the shortest path in the tree, between word and food. If the aspect else has more than one word, we take the minimum of the tree-based distances between a textual word and all the aspect integrant words. In the unusual case where more than one dependency trees are available in the context, manually we set the distance between the aspect term and the textual words in other trees to a constant, i.e. Half of the phrase length.

For a better demonstration of the suggested method, an instance is depicted in Figure 2. With the above-illustrated methods, the series of tree-based distances for all words in the sentence regarding the aspect term brass is spotted below the words in figure 2. The dependency proximity weights of the phrase are then allocated as:

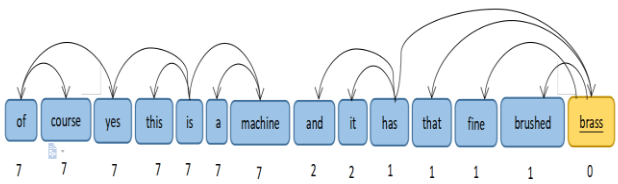


Figure 2. Dependency distance with respect to brass.

5. IMPLEMENTATION OF PWCN

In comparison with the benefits of word-level features, Aspect-based sentiment classification with phrase-level features has been depicted more efficiently. So, we are motivated to suggest a proximity-weighted convolution, which is necessarily 1-dimensional convolution with a length-l kernel, i.e. l-gram. Varied from the actual definition of convolution, the proximity-weighted convolution allocates proximity weight before convolution computation. The proximity-weight allocation process is formalized and where the formula represents the proximity weighted depiction of the i-th word in the sentence. Adding to that, we zero-pad the sentence to assure the convolution results in a series of the same length as the input. The convolution process includes the features that are drawn out by the convolution layer.

While few output features of a convolution layer are awaited to be explanatory for classification, we select the most wanted and important feature $q_s \in R^{2d}$ through a 1- dimension max-pooling layer with a kernel of length n, such that the most eminent feature vector q_s is given to a fully connected layer, following that a softmax normalization to acquire the distribution $y \in R^d$ over the deciding space on d_p-way sentiment polarity.Our model is tutored by the standard gradient descent algorithm, with cross-entropy loss as loss and L2 regularization.

6. DATASETS AND METHODS

We run the experiments on touchstone datasets from SemEval2014. This dataset has reviews and comments from two genres: Gadget and Restaurant, respectively.

In every experiment we perform, 300-dimensional GloVe is attached to begin the word embedding. All model parameters of our model began with the uniform distribution. The hidden state vectors' dimensionality is marked to 300. We then utilize Adam as the optimizer with 0.001 as the learning rate. The batch size is 64 and the coefficient of L2regularization is 10-5. We take on Macro-averaged F1 and Accuracy as the evaluation metrics. Adding to that, the length of the n-gram is marked up to 35.

Figure 3. Screenshot of Sample Datasets (SemEval 2014).

6.1. MODEL COMPARISON

An extensive comparison is taken place between our proposed model, i.e., PWCN with dependency proximity (PWCN-Dep), in opposition to various state-art baseline models, as described below:

TNet-LF - attaches Context-Preserving Transformation to secure and build up the instructive part of the text. It also gets benefitted from a multi-layer architecture.

LR - with L1 regularization can not only prevent over-tuning but also has a feature selection function.

SVM - has high versatility and high classification accuracy, and is suitable for binary classification problems.

IAN - trains attention in between aspect and its textual words in an interactive manner with two LSTM's.

RAM - will take up the external memory as hidden state vectors present in the text and tries Gated Recurrent Unit (GRU) structure to multi-hop attention. The top-most depiction is used to predict sentiments.

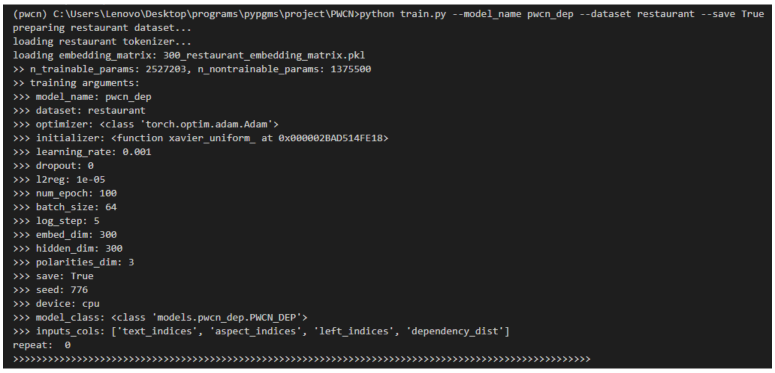
LSTM - only utilizes the hidden state vectors that are present at the end to predict the polarity.

Bi-LSTM - is a bidirectional recurrent neural network composed of two LSTMs in opposite directions. The outputs of the two LSTMs are combined to represent text functions and used for emotion classification.

CNN - The MLL algorithm based on neural network.

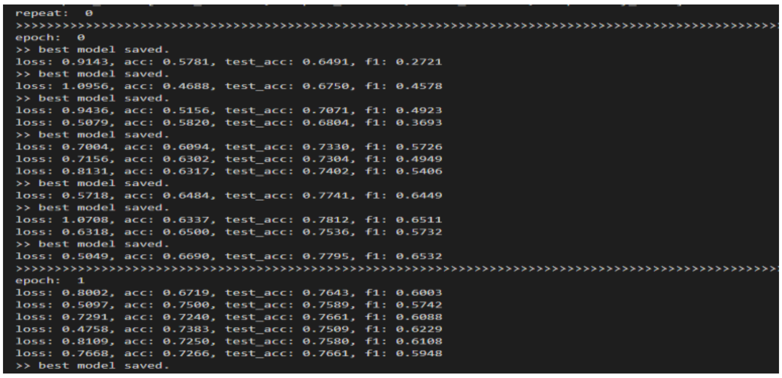
6.2. EXPERIMENTAL RESULTS

The results illustrate the basic efficiency of PWCN, which greatly out-runs LSTM, IAN and RAM and also accomplishes some great improvement over TNet-LF, which is one of the best-executing baseline models under comparison. Among the syntactic construction of sentences that are caught by the PWCN model, dependency proximity brings out more advantages regarding overall performance with constantly having greater Macro-F1 scores on datasets. The outputs also strengthen our argument that n-gram information is crucial for extracting features.



```
(pwcn) C:\Users\Lenovo\Desktop\programs\pypgms\project\PWCN>python train.py --model_name pwcn_dep --dataset restaurant --save True
preparing restaurant dataset...
loading restaurant tokenizer...
loading embedding_matrix: 300_restaurant_embedding_matrix.pkl
>> n_trainable_params: 2527203, n_nontrainable_params: 1375500
>> training arguments:
>>> model_name: pwcn_dep
>>> dataset: restaurant
>>> optimizer: <class 'torch.optim.adam.Adam'>
>>> initializer: <function xavier_uniform_ at 0x000002BAD514FE18>
>>> learning_rate: 0.001
>>> dropout: 0
>>> l2reg: 1e-05
>>> num_epoch: 100
>>> batch_size: 64
>>> log_step: 5
>>> embed_dim: 300
>>> hidden_dim: 300
>>> polarities_dim: 3
>>> save: True
>>> seed: 776
>>> device: cpu
>>> model_class: <class 'models.pwcn_dep.PWCN_DEP'>
>>> inputs_cols: ['text_indices', 'aspect_indices', 'left_indices', 'dependency_dist']
repeat:  0
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

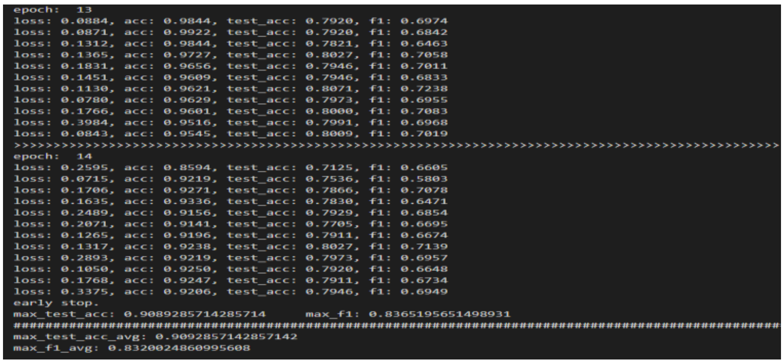
Figure 4. Training Datasets (Gadget and Restaurant).



```
repeat:  0
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
epoch:  0
>> best model saved.
loss: 0.9143, acc: 0.5781, test_acc: 0.6491, f1: 0.2721
>> best model saved.
loss: 1.0956, acc: 0.4688, test_acc: 0.6750, f1: 0.4578
>> best model saved.
loss: 0.9436, acc: 0.5156, test_acc: 0.7071, f1: 0.4923
loss: 0.5079, acc: 0.5820, test_acc: 0.6804, f1: 0.3693
>> best model saved.
loss: 0.7004, acc: 0.6094, test_acc: 0.7330, f1: 0.5726
loss: 0.7156, acc: 0.6302, test_acc: 0.7304, f1: 0.4949
loss: 0.8131, acc: 0.6317, test_acc: 0.7402, f1: 0.5406
>> best model saved.
loss: 0.5718, acc: 0.6484, test_acc: 0.7741, f1: 0.6449
>> best model saved.
loss: 1.0708, acc: 0.6337, test_acc: 0.7812, f1: 0.6511
loss: 0.6318, acc: 0.6500, test_acc: 0.7536, f1: 0.5732
>> best model saved.
loss: 0.5049, acc: 0.6690, test_acc: 0.7795, f1: 0.6532
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
epoch:  1
loss: 0.8002, acc: 0.6719, test_acc: 0.7643, f1: 0.6003
loss: 0.5097, acc: 0.7500, test_acc: 0.7589, f1: 0.5742
loss: 0.7291, acc: 0.7240, test_acc: 0.7661, f1: 0.6088
loss: 0.4758, acc: 0.7383, test_acc: 0.7509, f1: 0.6229
loss: 0.8109, acc: 0.7250, test_acc: 0.7580, f1: 0.6108
loss: 0.7668, acc: 0.7266, test_acc: 0.7661, f1: 0.5948
>> best model saved.
```

Figure 5. Best model is saved while Training datasets.

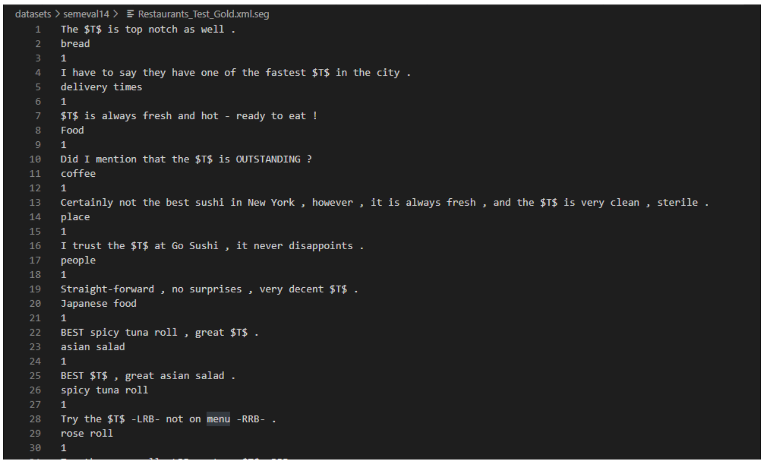
Furthermore, it is fascinating to notice that PWCN-based methodologies with unique syntactic information out-runs the other models that have the combination of syntactical and semantic information. As this showcases the majority of attaching syntactical dependency information to using semantic information, we furthermore guess that the attention mechanism will incorrectly supply term dependencies therefore it greatly damages the accurate decisions of PWCN.



```
epoch:  13
loss: 0.0884, acc: 0.9844, test_acc: 0.7920, f1: 0.6974
loss: 0.0871, acc: 0.9922, test_acc: 0.7920, f1: 0.6842
loss: 0.1312, acc: 0.9844, test_acc: 0.7821, f1: 0.6463
loss: 0.1365, acc: 0.9727, test_acc: 0.8027, f1: 0.7058
loss: 0.1831, acc: 0.9656, test_acc: 0.7946, f1: 0.7011
loss: 0.1451, acc: 0.9609, test_acc: 0.7946, f1: 0.6833
loss: 0.1130, acc: 0.9621, test_acc: 0.8071, f1: 0.7238
loss: 0.0780, acc: 0.9629, test_acc: 0.7973, f1: 0.6955
loss: 0.1766, acc: 0.9601, test_acc: 0.8000, f1: 0.7083
loss: 0.3984, acc: 0.9516, test_acc: 0.7991, f1: 0.6968
loss: 0.0843, acc: 0.9545, test_acc: 0.8009, f1: 0.7019
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
epoch:  14
loss: 0.2595, acc: 0.8594, test_acc: 0.7125, f1: 0.6605
loss: 0.0715, acc: 0.9219, test_acc: 0.7536, f1: 0.5803
loss: 0.1706, acc: 0.9271, test_acc: 0.7866, f1: 0.7078
loss: 0.1635, acc: 0.9336, test_acc: 0.7830, f1: 0.6471
loss: 0.2489, acc: 0.9156, test_acc: 0.7929, f1: 0.6854
loss: 0.2071, acc: 0.9141, test_acc: 0.7705, f1: 0.6695
loss: 0.1265, acc: 0.9196, test_acc: 0.7911, f1: 0.6674
loss: 0.1317, acc: 0.9238, test_acc: 0.8027, f1: 0.7139
loss: 0.2893, acc: 0.9219, test_acc: 0.7973, f1: 0.6957
loss: 0.1050, acc: 0.9250, test_acc: 0.7920, f1: 0.6648
loss: 0.1768, acc: 0.9247, test_acc: 0.7911, f1: 0.6734
loss: 0.3375, acc: 0.9206, test_acc: 0.7946, f1: 0.6949
early stop.
max_test_acc: 0.9089285714285714     max_f1: 0.8365195651498931
####################################################################################################
max_test_acc_avg: 0.9092857142857142
max_f1_avg: 0.8320024860995608
```

Figure 6. Calculation of Accuracy and F1 scores while Testing datasets.

To have clarity and to get to know about the effect that proximity weight has brought, we organize and regulate a case on an instance. More particularly, we can view the weights given by dependency proximity in PWCN-Dep along with its predictions as mentioned in Figure 7.


datasets > semeval14 > Restaurants_Test_Gold.xml.seg

```
The $T$ is top notch as well .
bread
1
I have to say they have one of the fastest $T$ in the city .
delivery times
1
$T$ is always fresh and hot - ready to eat !
Food
1
Did I mention that the $T$ is OUTSTANDING ?
coffee
1
Certainly not the best sushi in New York , however , it is always fresh , and the $T$ is very clean , sterile .
place
1
I trust the $T$ at Go Sushi , it never disappoints .
people
1
Straight-forward , no surprises , very decent $T$ .
Japanese food
1
BEST spicy tuna roll , great $T$ .
asian salad
1
BEST $T$ , great asian salad .
spicy tuna roll
1
Try the $T$ -LRB- not on menu -RRB- .
rose roll
1
```

Figure 7. Testing of datasets (PWCN-Dep).

We can also notice that the present attention mechanisms make incorrect decisions on which textual words elaborately showcases food, as both kind of proximity weight in our model can manage this error clearly, which is what we expect.

7. CONCLUSION AND FUTURE WORKS

Earlier methodologies of using aspect information for the aspect-based sentiment classification from the semantic viewpoint, as the syntactic relationships between the aspect and its textual words are basically avoided. In this paper, we have developed a framework that attaches n-gram information and syntactic dependency between aspect and textual words into a usable and approachable model. Experimental results have illustrated the accuracy of our suggested models is 90% (as mentioned in Figure 6) and is all way compatible (i.e., syntax-free) and proposed that syntactic dependency is more advantageous for aspect-based sentiment classification than semantic relatedness.

We strongly believe that it is a favourable direction to jump into solid instances to examine the difference that lies between PWCN models and attention-based models to accomplish a cavernous understanding of where syntactical dependencies dumbfound semantic relatedness.

REFERENCES

[1] S. M. Al-Ghuribi, S. A. Mohd Noah and S. Tiun, "Unsupervised Semantic Approach of Aspect-Based Sentiment Analysis for Large-Scale User Reviews," in IEEE Access, vol. 8, pp. 218592-218613, 2020, doi: 10.1109/ACCESS.2020.3042312.
[2] K. Xu, H. Zhao and T. Liu, "Aspect-Specific Heterogeneous Graph Convolutional Network for Aspect-Based Sentiment Classification," in IEEE Access, vol. 8, pp. 139346-139355, 2020, doi: 10.1109/ACCESS.2020.3012637.
[3] K. Aurangzeb, N. Ayub and M. Alhussein, "Aspect Based Multi-Labeling Using SVM Based Ensembler," in IEEE Access, vol. 9, pp. 26026-26040, 2021, doi: 10.1109/ACCESS.2021.3055768.
[4] M. Shams, N. Khoshavi and A. Baraani-Dastjerdi, "LISA: Language-Independent Method for Aspect-Based Sentiment Analysis," in IEEE Access, vol. 8, pp. 31034-31044, 2020, doi: 10.1109/ACCESS.2020.2973587.
[5] C. R. Aydin and T. Güngör, "Combination of Recursive and Recurrent Neural Networks for Aspect-Based Sentiment Analysis Using Inter-Aspect Relations," in IEEE Access, vol. 8, pp. 77820-77832, 2020, doi: 10.1109/ACCESS.2020.2990306.
[6] A. Ishaq, S. Asghar and S. A. Gillani, "Aspect-Based Sentiment Analysis Using a Hybridized Approach Based on CNN and GA," in IEEE Access, vol. 8, pp. 135499-135512, 2020, doi: 10.1109/ACCESS.2020.3011802.
[7] Y. Han, M. Liu and W. Jing, "Aspect-Level Drug Reviews Sentiment Analysis Based on Double BiGRU and Knowledge Transfer," in IEEE Access, vol. 8, pp. 21314-21325, 2020, doi: 10.1109/ACCESS.2020.2969473.
[8] F. Yin, Y. Wang, J. Liu and L. Lin, "The Construction of Sentiment Lexicon Based on Context-Dependent Part-of-Speech Chunks for Semantic Disambiguation," in IEEE Access, vol. 8, pp. 63359-63367, 2020, doi: 10.1109/ACCESS.2020.2984284.
[9] K. Cheng, Y. Yue and Z. Song, "Sentiment Classification Based on Part-of-Speech and Self-Attention Mechanism," in IEEE Access, vol. 8, pp. 16387-16396, 2020, doi: 10.1109/ACCESS.2020.2967103.

[10] K. Abdalgader and A. A. Shibli, "Experimental Results on Customer Reviews Using Lexicon-Based Word Polarity Identification Method," in IEEE Access, vol. 8, pp. 179955-179969, 2020, doi: 10.1109/ACCESS.2020.3028260.
[11] Jagdale R.S., Shirsat V.S., Deshmukh S.N. (2019)" Sentiment Analysis on Product Reviews Using Machine Learning Techniques," In: Mallick P., Balas V., Bhoi A., Zobaa A. (eds) Cognitive Informatics and Soft Computing. Advances in Intelligent Systems and Computing, vol 768. Springer, Singapore. https://doi.org/10.1007/978-981-13-0617-4_61.
[12] F. Iqbal et al., "A Hybrid Framework for Sentiment Analysis Using Genetic Algorithm Based Feature Reduction," in IEEE Access, vol. 7, pp. 14637-14652, 2019, doi: 10.1109/ACCESS.2019.2892852.
[13] Kumar, R., Pannu, H.S. & Malhi, A.K. "Aspect-based sentiment analysis using deep networks and stochastic optimization," Neural Comput & Applic 32, 3221–3235 (2020). https://doi.org/10.1007/s00521-019-04105-z.
[14] Abdi, A., Shamsuddin, S., Hasan, S., & Piran, J. (2019) "Deep learning-based sentiment classification of evaluative text based on Multi-feature fusion," Inf. Process. Manag., 56, 1245-1259.
[15] Rezaeinia, S.M., Rahmani, R., Ghodsi, A., & Veisi, H. (2019). "Sentiment analysis based on improved pre-trained word embeddings," Expert Syst. Appl., 117, 139-147.
[16] E. Zuo, H. Zhao, B. Chen and Q. Chen, "Context-Specific Heterogeneous Graph Convolutional Network for Implicit Sentiment Analysis," in IEEE Access, vol. 8, pp. 37967-37975, 2020, doi: 10.1109/ACCESS.2020.2975244.
[17] Meskele, D., & Frasincar, F. (2019) "ALDONA: a hybrid solution for sentence-level aspect-based sentiment analysis using a lexicalised domain ontology and a neural attention model," Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing.
[18] Tao, J., Fang, X. "Toward multi-label sentiment analysis: a transfer learning-based approach," J Big Data 7, 1 (2020). https://doi.org/10.1186/s40537-019-0278-0.
[19] Z. Kastrati, A. S. Imran and A. Kurti, "Weakly Supervised Framework for Aspect-Based Sentiment Analysis on Students' Reviews of MOOCs," in IEEE Access, vol. 8, pp. 106799-106810, 2020, doi: 10.1109/ACCESS.2020.3000739.
[20] Da'u, Aminu, et al. "Recommendation system exploiting aspect-based opinion mining with deep learning method." Information Sciences 512 (2020): 1279-1292.
[21] Wu, Guoqiang et al. "Joint Ranking SVM and Binary Relevance with Robust Low-Rank Learning for Multi-Label Classification." Neural networks: the official journal of the International Neural Network Society 122 (2020): 24-39.
[22] Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Syntax-Aware Aspect-Level Sentiment Classification with Proximity-Weighted Convolution Network. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19). Association for Computing Machinery, New York, NY, USA, 1145–1148. DOI: https://doi.org/10.1145/3331184.3331351.