Approaching the Summarization of Online News Article Discussions with Topic Clusters based on Deep Neural Networks

Marvin Bornstein Willi Gierke Tobias Nack
Hasso Plattner Institute
Prof.-Dr.-Helmert-Str. 2-3
Potsdam, Germany
{firstname.lastname}@student.hpi.de

ABSTRACT

We propose an approach to select the most suitable comments summarizing discussions of online news articles. Our approach focuses on finding different topic clusters within a single discussion, which we can then summarize by finding representative comments. This ensures that the final summary covers a wider variety of topics discussed. The topic clusters are created by using the density based clustering approach *DBSCAN* on embeddings generated for the text of every comment using a neural network based on bi-directional LSTMs. We train the network using a triplet loss, forcing the embeddings of comments from the same article closer together compared to comments from different articles. With this approach, we compare similarly to the commercial summarization tool *SMMRY* judging by a user study with 12 participants.

KEYWORDS

Natural Language Processing, Discussion Summarization, Triplet Loss, Clustering

1 INTRODUCTION

Online news have become the most important source of information for the majority of citizens [17]. These news sources often offer a comment section accompanying their articles for users to engage in further discussions. Especially for large international news sources and highly polarizing topics, these discussions can become too long to effectively comprehend for most readers, which makes them less likely to take part. Therefore it would be helpful to automatically provide a summary for the comment section, supporting a reader in their decision to read further and contribute their own thoughts. The generation of such a summary for a collection of comments differs from other more common summarization tasks on continuous texts such as news articles or scientific papers. The variety of comments written by different authors is hereby more likely to cover broader range of topics. Also, conversations between users are likely to experience a topic drift. To capture this, we propose an approach to cluster the comments into different topic groups using unsupervised machine learning techniques. These clusters can then be summarized separately. The machine learning techniques are inspired by current state of the art literature for text mining, language modeling and other deep learning tasks. Our main contribution is the collection of the different techniques and creation of a coherent pipeline specifically orientated for online newspaper discussions.

2 RELATED WORK

Given a set of comments to an article, it is desired to generate a summary that preserves the most important information. Classical approaches are based on the relative importance of words in the document as measured by the TF-IDF. As an example, *LexRank* [5] finds the most important sentences by using the eigencentrality of a graph that is induced by the similarity matrix between the TF-IDF vectors of the sentences. Another well-known tool from the industry is *SMMRY*[1]. While the underlying algorithm has not been disclosed, the inventors state that it ranks sentences depending on the relative occurrences of the words contained in the sentences, which indicates that TF-IDF vectors are used as well. These approaches are intended to work with coherent text and are therefore not necessarily applicable to our use cases. Another disadvantage of these approaches is that the TF-IDF embedding embodies no semantic relationship between the words. Word embeddings like *FastText* [2] overcome this obstacle by encoding words with similar meaning in a similar vector representation. *FastText* breaks words into n-grams for training. Thus, words that are missing as a whole from the training data can also be embedded. The representation of a word is then the sum of its n-gram representations. While the embedding choice is an important preprocessing step, the actual learning algorithm is at least as important.

Recent advances in deep learning have shown promising results for learning representations of hierarchical data such as images and text. Thus, a large corpus of recent work exists that uses deep learning for natural language processing problems. For instance, convolutional neural networks (CNNs) [15] have shown excellent performance on various tasks on text such as sentence classification [11]. More recently, bidirectional long short-term memory networks (BiLSTM) [9] have been used to achieve exceptional results on tasks such as named entity recognition [10, 13]. To free architectures for natural language generation from their fixed-length internal representation constraint, attention has been introduced by Bahdanau et al. [1]. These concepts have been jointly used for natural language generation to e.g. sum up a headline and a sentence of an article in a shorter sentence [8, 16, 20, 22, 24]. Unfortunately, the algorithms need a large corpus of labeled training data which is unfeasible in our case. Therefore, we decided to only show the most informative comments to the user instead of generating natural language.

To display only the best comments, it is necessary to identify the properties of such comments. It has been shown that editor picks, discussion comments highlighted by editors, are e.g. critical

[1] https://smmry.com/

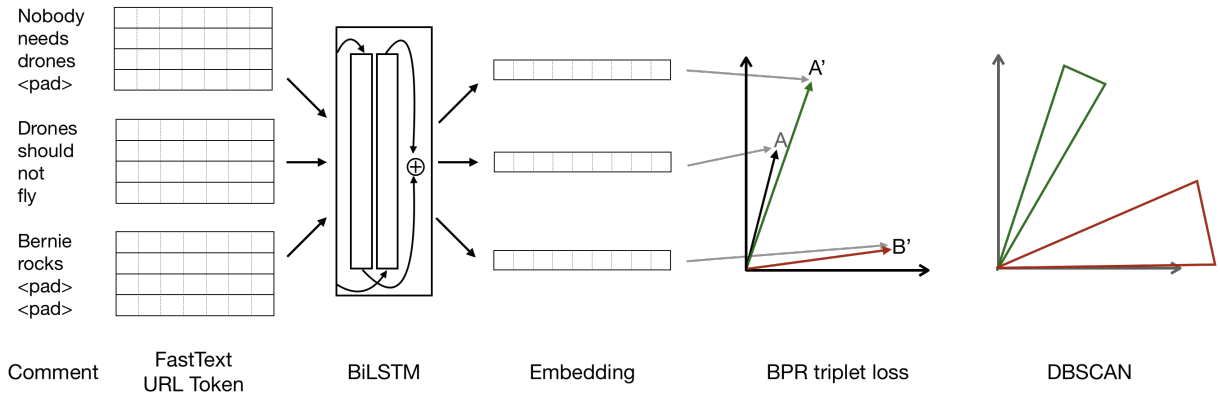

Figure 1: Schematic pipeline of our algorithm. Sequence encoding of comments, which is trained to minimize a triplet loss and clustered using DBSCAN.

and address different topics [4, 12]. These could be detected via handcrafted features such as discourse indicators [14]. However, these indicators are rarely used in comments and as such do not scale. Farrell et al. [7] incorporate hand-crafted features and the discussion hierarchy to create a summary by selecting the most salient comments. Due to the manually created features, the scalability of the approach is very limited. As can be seen, unsupervised discussion summarization using word embeddings and representation learning have not achieved much attention so far.

3 ALGORITHM

In order to deal with the heterogeneous nature of a discussion, we propose a pipeline that can be seen in Figure 1 which finds homogeneous clusters within a discussion. We can then use a summarization algorithm to find the central comment of a cluster to represent it. Our pipeline starts by finding an appropriate embedding of every comment in a representation space using a sequential machine learning model. These representations can then be used together with classical clustering approaches to find the homogeneous groups in a discussion.

3.1 Comment representation

Word representation models, such as *FastText* [2], allow us to embed word fragments, words or sequences of words into an k-dimensional, real representation vector. These vectors are optimized to encode word analogies and similarities and are trained on generic text data. Encoding every comment in a discussion that way does not necessarily capture the domain specific topics or stance.

Thus, we extend this word level embedding to a comment level embedding by feeding the *FastText* embeddings trained on a large online news comment corpus into a sequence model, e.g. a convolutional or recurrent network. For our final implementation, we decided to use a BiLSTM model, which yielded the best results in our later evaluation (cf. outlier detection in Section 4.1).

Since we do not have a labeled data set for our task, we turn the unsupervised problem into a semi-supervised one and optimize our sequence model to capture topic or discussion-specific information. The objective is to encode comments to the same discussion closer to comments from different discussions.

$$\mathcal{L}_{\text{bpr}} = 1 - \sigma(\mathbf{a}^T\mathbf{p} - \mathbf{a}^T\mathbf{n}) \tag{1}$$

We minimize the triplet loss [18, 21] in Equation 1, where $\mathbf{a}, \mathbf{p}, \mathbf{n} \in \mathbb{R}^k$ are the *anchor*, *positive* and *negative* k-dimensional representation vectors of a comment and σ is the sigmoid function. Representation vectors are the element-wise average of the last outputs of the forward and backward LSTM. The anchor and positive example are drawn from the same discussion, the negative from another. This *BPR* triplet loss operates on similarities comparable to cosine similarity. This means that the norm of the representations is not constrained, but the representations pairwise angles are. We experimented with an Euclidean distance in the loss, but it would always collapse the embedding space to a singularity during training. We believe this is due to the high dimensional representation space, in which Euclidean distances do not differ much. It is yet to show whether a task specific triplet mining strategy would improve training.

Intuitively, the model distinguishes comments by topic or opinion and separates them into homogeneous groups. Every generic comment that fits multiple discussions may fall into a *noise* region in the representation space. We assume this separation to generalize to comments within a discussion.

3.2 Discussion groups

Using the introduced comment embeddings that naturally group comments, we can use traditional clustering algorithms to identify these groups. Note that our objective in Equation 1 does not minimize Euclidean distance. Thus we use the density-based algorithm *DBSCAN* [6] on pairwise cosine similarity to cluster comments within a discussion. *DBSCAN* allows to find a variable amount of clusters and identify outliers. This means that we account for comments that are too generic, like "Well done." or "First!", and the fact that topics can have varying amounts of subtopics or stances. We automatically optimize *DBSCAN*s hyper-parameter ϵ for every article to maximize the number of clusters. We achieved better clustering results when we reduced the dimensionality of our embeddings with PCA [23] before fitting *DBSCAN*. We think that this implies that our sequence model encodes redundant information

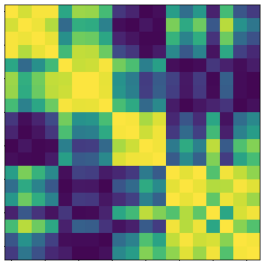

Figure 2: Cosine similarity matrix of discussion without noise comments and ordered by *DBSCAN* grouping. Clusters appear to be quite homogeneous, i.e. have few similarities across clusters.

which would worsen the performance of clustering approaches that generally prefer to work in low-dimensional space.

3.3 Group representative

The cosine similarity matrix in Figure 2 shows quite independent, homogeneous groups after clustering the comments. We choose cosine similarity to comply with the embedding space, which optimized the *BPR* triplet loss. Within each group, we assume the most central comment to be a good representative, which has been a successful approach in document summarization. Within each cluster, we compute the similarity matrix and score sentences by the eigencentrality, in accordance to Erkan and Radev [5]. Eigencentrality not only captures the local similarity of one comment to another, but also if their similar comments are itself very similar within the cluster. This results in a proper measure for centrality and a good way to score each comment's importance in the overall discussion.

3.4 Implementation details

We built our models in Python [19] by using the Keras library [3]. *FastText* embeddings were trained on our dataset with 50 embedding dimensions. We limited the sequence length of our BiLSTM to 100 words, zero-padded shorter comments and cut larger comments after 100 words. Our CNN model consists of 1D convolutions with kernel sizes of 3, 4 and 5. We generate 100 filter maps for each kernel size. A max-over-time pooling layer reduces the output to a 300 dimensional representation vector. This architecture is inspired by the work of Kim [11].

Our BiLSTM model averages the last output of both single LSTM models which have 124 cells each and *tanh* activation. We added a fully connected layer with ReLU activation to reduce the 124 dimensional output to 100 dimensions.

4 EVALUATION

We evaluated our approach on a dataset of around 61 million comments crawled from the English newspaper *The Guardian*. The comments are taken from articles published between 2005 and 2017. Our model was trained on the first 20 million comments and all other evaluation methods use the last 10000 comments to avoid testing our model on any before seen comments or articles. The methods that we use to evaluate our approach are an outlier detection and a user study. Both test different properties, namely the quality of the learned representation and the quality of the complete summary.

4.1 Outlier detection

This technique is based on artificially inserting comments from another article and checking whether our approach spots them as outliers or a cluster that is only coherent within itself. Since different articles are very likely to address different topics, we see a good performance in separating the comments of the articles as an indication of our approach actually separating by topics. We tested this using 100 different articles that were not part of the training set. Every article has between 25 and 250 comments and depending on the number of comments we add between 5 and 50 randomly selected comments from one different article of this set as artificial noise. One should note that the original articles also have comments that should be classified as outliers, if they are for example either completely off-topic or short answers to other people without any further content. That means it is not necessarily desirable to have no comments from the original article found as outliers, which is also in line with our results. In comparison, the chance of two articles with the same topics in the discussion being combined is relatively small, which makes it desirable for the noise comments to be classified as outliers with a rate as close to 100% as possible.

Approach	Real comments	Noise comments
Baseline Model	45%	66%
CNN Model	31%	72%
BiLSTM Model	**28%**	**84%**

Table 1: Percentage of comments classified as outliers

Table 1 shows the results of this experiment comparing three different models giving the mean percentage of outliers for all tested combinations of article and noise. The results of the different combinations do not significantly deviate from each other. The three models only differ in the way the embedding of a comment is generated, both the word embedding beforehand and density based clustering afterwards are consistent with our pipeline. The baseline model uses the average of all word embeddings in the comment as representation for the clustering. Both the CNN model as well as the BiLSTM model were trained as previously described using the BPR triplet loss. For all three models we can see that comments inserted as noise are more likely to be classified as outliers compared to the comments of the original article. This shows that all three models already include general information about the topic. Both embeddings with neural networks conclusively beat out the baseline model that just uses the information from the word embedding indicating that training with the BPR triplet loss has a positive effect on our task. We can also observe that the performance for the CNN model and BiLSTM model is relatively similar for the comments of the original article. However, the BiLSTM model is significantly better in correctly spotting the noise comments as outliers, which is why we decided to choose the BiLSTM model in our final approach and use it for the user study. We think that

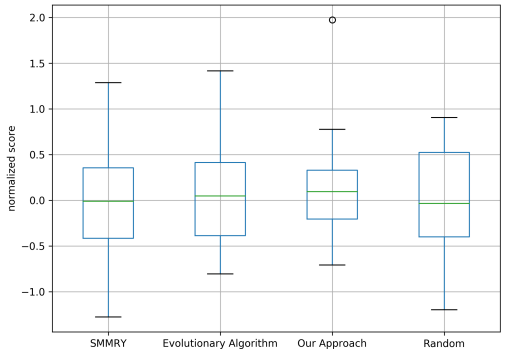

Figure 3: Boxplot of the normalized ratings for every approach. The median scores do not differ much compared to the deviation.

the CNN model might perform better if further techniques such as attention would have been used.

4.2 User study

As a second technique to evaluate the summarization quality of our approach we ran a user study. The participants evaluated the summarization quality of four different approaches giving us the opportunity to compare them relative to each other. Apart from our described approach and the one of another tutorial group using an evolutionary algorithm we included *SMMRY* and selecting random comments as baselines. To save the subject group reading the entire articles and discussion sections, we only displayed the four proposed summaries. An accompanying guideline describes appropriate summaries as e.g. short, expressive, serious and understandable as inspired by Diakopoulos [4], Kolhatkar and Taboada [12].

In total we collected 902 scores for 43 different articles from 12 computer science students. We normalized the scores for any participant to better compare people that generally score with higher or lower grades and excluded all articles that had been labeled by less than three participants. All four approaches are scored relatively similarly with neither getting a real edge over the other competitors. Figure 3 shows that the results of all approaches can vary relatively widely depending on the underlying article with our approach showing the most stable scores. To gain further insights, we manually analyzed the created summaries of articles on which we either performed exceptionally well or very badly. We could not gain much insights from the articles on which we performed well. However, we found that the length of badly rated summaries varies greatly from our average summary length of around 261 words. Summaries that have been rated badly are either very short with around 30 to 40 words or very long with between 650 to 850 words.

5 CONCLUSION

In this work, we have addressed the problem of summarizing an online discussion by selecting the most suitable comments in an unsupervised manner. Technically, we have proposed an end-to-end trainable approach based on several state of the art components. Qualitative experiments have shown that the approach is able to learn an embedding that separates discussions with different topics. However, the results of our user study are not convincing, which we think is most likely related to the choice of comments representing the different clusters. This indicates that in future work one should concentrate on finding better techniques or heuristics for choosing appropriate comments. After the analysis of our user study, we believe that relatively easy approaches such as limiting the maximum and minimum length of the chosen representatives could already greatly improve the quality of our summaries.

REFERENCES

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0473, 2014.

[2] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[3] F. Chollet et al. Keras, 2015.

[4] N. Diakopoulos. Picking the NYT Picks : Editorial Criteria and Automation in the Curation of Online News. 2015.

[5] G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479, 2004.

[6] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, 1996.

[7] R. Farrell, P. G. Fairweather, and K. Snyder. Summarization of discussion groups. In *Proceedings of the 10th international conference on Information and knowledge management - CIKM'01*, page 532. ACM Press, 2001.

[8] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional Sequence to Sequence Learning. *CoRR*, abs/1705.03122, 2017.

[9] A. Graves, S. Fernández, and J. Schmidhuber. Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition. Technical report, 2005.

[10] Z. Huang, B. Research, W. Xu, and K. Y. Baidu. Bidirectional LSTM-CRF Models for Sequence Tagging. Technical report, 2015.

[11] Y. Kim. Convolutional Neural Networks for Sentence Classification. *CoRR*, abs/1408.5882, 2014.

[12] V. Kolhatkar and M. Taboada. Using New York Times Picks to Identify Constructive Comments. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*, pages 100–105. Association for Computational Linguistics, 2017.

[13] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. Neural Architectures for Named Entity Recognition. Technical report, 2016.

[14] J. Lawrence and C. Reed. Combining Argument Mining Techniques. In *Proceedings of the 2nd Workshop on Argumentation Mining*, 2015.

[15] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[16] R. Nallapati, B. Zhou, C. N. dos Santos, C. Gulcehre, and B. Xiang. Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. *CoRR*, abs/1602.06023, 2016.

[17] N. Newman, R. Fletcher, A. Kalogeropoulos, D. A. L. Levy, and R. Kleis Nielsen. Reuters Institute for the Study of Journalism / Digital News Report 2018.

[18] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[19] G. Rossum. Python Reference Manual. Technical report, Amsterdam, The Netherlands, The Netherlands, 1995.

[20] A. M. Rush, S. Chopra, and J. Weston. A Neural Attention Model for Abstractive Sentence Summarization. *CoRR*, abs/1509.00685, 2015.

[21] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. *CoRR*, abs/1503.03832, 2015.

[22] A. See, P. J. Liu, and C. D. Manning. Get To The Point: Summarization with Pointer-Generator Networks. *CoRR*, abs/1704.04368, 2017.

[23] M. E. Tipping and C. M. Bishop. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society, Series B*, 61(3):611–622, 1999.

[24] W. Yin and H. Schütze. Attentive Convolution. *CoRR*, abs/1710.00519, 2017.