Text Classification using Capsule Routing

Jishnu Ray Chowdhury
jraych2@uic.edu
University of Illinois Chicago

1 ABSTRACT

In this work, we study and compare multiple capsule routing algorithms for text classification including dynamic routing, Heinsen routing, and capsule-routing inspired attention-based sentence encoding techniques like dynamic self-attention. Further, similar to some works in computer vision, we do an ablation test of the capsule network where we remove the routing algorithm itself. We analyze the theoretical connection between attention and capsule routing, and contrast the two ways of normalizing the routing weights. Finally, we present a new way to do capsule routing, or rather iterative refinement, using a richer attention function to measure agreement among output and input capsules and with highway connections in between iterations.

2 INTRODUCTION

Capsules are groups of neurons which represent a specific type of entity. Capsule Routing was originally proposed in Computer Vision [14, 19] to model the hierarchical and part-whole relationships among the entities (at different levels of hierarchy) within an image. These forms of relationships are also prevalent in the Natural Language domain. Consider the hidden states in a sequence as 'parts', and the class-entities or sentence representations as 'wholes', and thus classification and sentence-encoding tasks can be thought to require modeling of part-whole hierarchies. Indeed, there are researchers [24] who noted this and brought Capsule Routing into the Natural Language domain.

However, we lack a decent comparison (under comparable settings) of different capsule routing algorithms that have been proposed in NLP. Furthermore, there are new developments in Computer Vision regarding capsule routing [3, 17] which are yet to be investigated for NLP. We empirically compare different existing capsule routing algorithms on some text classification datasets (AAPD, Reuters) under comparable settings. Inspired from the recent developments in computer vision, we also try a special ablation test where we keep certain portions of a capsule network but remove the routing mechanism. We also explore some new modifications to capsule routing algorithms, especially a highway connection and a richer attention-based function to measure agreement among output and input capsules. The details of our models are further discussed in Section 4.

3 RELATED WORKS

While the idea behind capsules [12, 13] is quite old, a concrete dynamic capsule routing algorithm was first introduced in [19]. Hinton et al. [14] improved it by proposing an Expectation-Maximization based routing algorithm for capsules. In NLP, Yang et al. [24] were the first to apply capsule routing for text classification. Feng et al. [8] explored capsule routing in multimodal settings. Cheng et al. [4] used the routing-by-agreement (used for capsule routing) algorithm to model multiple perspectives for natural language inference. Zheng et al. [29] combined attention mechanism with hierarchical dynamic capsule routing. Du et al. [7] used capsule network with interactive attention for aspect-level semantic classification. Du et al. [6] accounted for the mixture of different word-senses in classic word embeddings through capsule routing. Xiao et al. [23] proposed MCapsNet which uses capsule routing for multi-task learning. Zhang et al. [26, 27] used attention based capsule network for relation extraction. Zhao et al. [28] modified capsule routing to make it more scalable and reliable for NLP applications. Yoon et al. [25] used a dynamic iterative self-attention based aggregation method for sentence encoding inspired from the dynamic capsule routing algorithm. Heinsen [10] introduced a new routing algorithm (Heinsen Routing). Heinsen Routing is still based on EM-routing but it also has an extra 'D-step'. Essentially it is based on the notion that "output capsules should benefit from the input data they use, and lose benefits from any input data they ignore," Capsule routing had also been used to improve Transformers [22]. Gu and Feng [9], Li et al. [16] used it for combining information from different attention heads. Dou et al. [5] used the iterative capsule routing algorithm for deep layer aggregation in Transformers. Therefore, our investigation can also contribute to the further development of Transformers.

4 METHODOLOGY

We initially tried to create a framework similar to Yang et al. [24] for the routing algorithms but we were unable to run it in a limited resource setting when tackling sequences of longer lengths. Therefore, we settled for a simpler alternative which we discuss below.

4.1 Convolutional Layer

For the sake of comparability, we keep the overall framework similar for all models. The first layer of the framework is a convolutional layer. Similar to Kim [15], we use a single CNN layer where three parallel convolutional operations take place with different kernel sizes. In our settings, we use kernel sizes of 3, 4, and 5. For each kernel size, we use 128 output filters. The CNN layer can extract different n-gram features which can be effective for classification tasks. Followed by the CNN, we either have a pooling layer, capsule network, or an attention layer depending on the model we are using. For the basic CNN baseline, we apply global max pooling (over the temporal dimension) on the output filters for each kernel size separately and then we concatenate them. We then pass the vector representation through multiple classifiers for the classification task. In the "CNN + attention" baseline, instead of max pooling, we concatenate the outputs from each kernel size along the feature dimension, and use an attention mechanism [1] along the temporal dimension. Essentially we use a linear layer to score each hidden

state representation in each position of the sequence. The score is normalized with softmax, and the final result is the weighted average of the token representations based on the normalized scores as weights. In the subsequent sections, we discuss what we do when we use a capsule (or a capsule-inspired) network instead of max-pooling or attention.

4.2 Primary Capsule Layer

The target of this layer is to transform the output of the CNN layer into capsules. In our work, we treat each token in each position of the sequence as a capsule. For the purpose of the transformation, we simply use a non-linear layer to project the CNN output into a different dimension (64, in our case). We treat the outputs as input capsules to the next layer. However, some of the details for this layer vary depending on the routing algorithm that we are using. For dynamic routing, we specifically use the squashing function (as described in [19]) to make sure that the length of the capsule-vectors does not exceed 1. In dynamic routing, we treat the length of the capsule-vectors as the probability of the existence of the entity that they represent. Heinsen Routing [10], on the other hand, requires us to maintain an extra scalar for each capsule to represent its probability. So for Heinsen Routing, we use another non-linear layer with a sigmoid activation function to compute the probability of each capsule. For any routing algorithm which does not use the length of the vector to represent its probability, we use the GELU activation function [11] for non-linearity.

Algorithm 1 CoDA Routing algorithm

procedure CoDARouting($\hat{u}_{j|i}, r, l$)
  $b_{j|i} \leftarrow 0$
  $a_{j|i} \leftarrow normalize(b_{j|i})$
  for $iter$ `in 0 to` r **do**
    $\hat{v}_j \leftarrow g(\sum_i (a_{j|i}\hat{u}_{j|i}))$ ▷ g is an activation function
    if $iter == 0$ **then**
      $v_j \leftarrow \hat{v}_j$
    else
      $\alpha \leftarrow \sigma(W\hat{v}_j + b)$ ▷ α is scalar. σ is the sigmoid
      $v_j \leftarrow \alpha \cdot \hat{v}_j + (1 - \alpha) \cdot v_j$ ▷ highway connection
    end if
    $e_{j|i} = tanh(< f_1(v_j), f_2(\hat{u}_{j|i}) >)$ ▷ fs are affine layers
    $n_{j|i} = \sigma(-||f_1(v_j) - f_2(\hat{u}_{j|i})||_{l_1})$
    $a_{j|i} = e_{j|i} \odot n_{j|i}$
    return v_j

4.3 Fully Connected Capsule Layer

Finally, in this layer, we route the input capsules (which are outputs from the primary capsule layer) into the output capsules. There are several routing structures that can be used. For most of our models, we treat each output capsul as the representation for each class. We can then use the probability of that class-capsule as the probability for the class or we can use a sigmoid layer on top of the class representation to get the probability. For dynamic routing, we use the former, for other cases we use the later. Alternatively, in some cases, we also explore routing the input capsules to a single capsule which can be treated as a sentence vector representation. The sentence vector can be then be used for classification. We describe more about in which cases we use sentence encoding in section 4.4.

For the routing process to take place, the first step is to compute the votes for each output capsule from each input capsule. Let $\hat{u}_{j|i}$ represent the vote from capsule i to capsule j. The dimension of $\hat{u}_{j|i}$ has to be the same as the dimension of the output capsules. Each capsule i is fed to j different linear layers to compute the votes for j (we share the parameters for every input capsules as we have a variable number of input capsules dependent on the sequence length).

$$\hat{u}_{j|i} = W_j u_i + c_j$$

Where u_i is the representation for input capsule i, c_j is the bias term (scalar), and W_j is the weights of shape $d_{out} \times d_{in}$ where d_{out} is the dimension of the output capsule and d_{in} is the dimension of the input capsule. In our case, for both dimensions we use 64.

In addition, we also have $b_{j|i}$ which denotes the connection strength or routing co-efficient from input capsule i to output capsule j. $b_{j|i}$ is iteratively refined in each iteration of routing. Similar to Yang et al. [24], we also use a leaky-softmax and an orphan output capsule which is ignored in subsequent layers. The orphan capsule acts like a "trashbin" where unnecessary features (for example, those related to stopwords) can be routed to from the input capsules. Below, we discuss different types of routing algorithms that we use.

4.3.1 ***Dynamic Routing and Heinsen Routing.*** We experiment with both dynamic routing [19, 24] and Heinsen routing [10]. Heinsen routing can also be thought of as a better variant of EM-routing [14]. For the sake of brevity, we leave out the details of the algorithms from this report and suggest the reader to refer to the cited papers as needed.

4.3.2 ***Dynamic Self-Attention.*** Dynamic Self-Attention (DSA) [25] uses a routing like method over the votes to iteratively refine attention weights which are used to create a sentence representation through weighted average of the hidden state representations. We explore using DSA, both to create multiple "capsule" representations (using multiple sentence attentions) to represent different classes and also to create a single sentence vector representation.

4.3.3 ***No-Routing Algorithm.*** This is an ablation inspired by some of the recent developments [3, 17] in computer vision. Paik et al. [17] argued that we need a better routing algorithm. They empirically demonstrated on Computer Vision tasks that the current capsule network-based models do not improve with increasing routing iterations, and models without any routing iterations (just using random or uniform initialization of weights for capsule votes) even perform better. They showed that it holds true for a number of existing capsule routing algorithms. Chen et al. [3] further explored this line of work and proposed P-CapsNet where the authors removed the iterative routing algorithm altogether. They argued that much of the benefits from capsule routing may be from high-order tensor multiplications involved in the computation of votes for output capsules rather than the routing algorithm itself. Inspired from these discoveries, in our ablation we just keep the votes and remove the routing procedure. Instead, we compute the output

capsule representations as:

$$v_j = g(\sum_i \hat{u}_{j|i})$$

Where v_j is the output capsule j, g is an activation function (squash or GELU), $\hat{u}_{j|i}$ are the votes as discussed before.

4.3.4 ***CoDA Routing Algorithm.*** Besides all of them, we also develop our own modifications to the existing routing algorithms creating what we dub as CoDA Routing. The pseudo-code of CoDA Routing is described in 1. The two main differences in CoDA Routing are the addition of highway connections[20] and the use of Compositional De-Attention (CoDA) [21] to measure the agreement between the tentative output capsule v_j and the votes to update the connection weights. Noticing the issues with routing procedures and how more iterations are not always better [17], we add a highway connection in each iteration (see step 9 and 10 in 1). Furthermore, motivated to experiment with a richer agreement-based scoring method we try compositional de-attention (CoDA) instead of a simple dot product. Our exact implementation of CoDA-based-capsule-agreement-based computation of routing weights are given in steps 12 to 14 in 1. We also remove the consideration of previous $b_{j|i}$ weights when using the attention function. Taking some inspirations from Dynamic Self-Attention, we do not restrict ourselves with all the capsule terminologies and constraints. We do not anymore constrain the vectors such that their lengths represent their probabilities and we do not maintain any extra scalar to represent their probability.

4.4 Attention, Normalization, and Sentence Encoding

In 1, we notice a "normalize" function in step 3. Some form of normalization is always present in any of these routing algorithms. In this work, it is usually the leaky-Softmax function. But there are two ways to apply this normalization: along the axis of input capsules (input-normalization) or along the axis of output capsules (output-normalization). We discuss both of them along with their intuitive function and connection to attention below:

4.4.1 ***Input Normalization (IN).*** Input normalization with Softmax can be formally depicted as:

$$a_{j|i} = \frac{exp(b_{j|i})}{\sum_i exp(b_{j|i})}$$

This form of normalization is not standardly used by capsule networks. However, DSA and standard attention mechanism-based methods to create sentence representations use it. Intuitively, here, the input capsules compete for the attention of the output capsules. The same input capsule can be assigned very high (near 1) normalized weights for all the output capsules, but multiple input capsules cannot be assigned very high normalized weights (near 1) for any given output capsule. Thus, in this formulation, it is also naturally possible to take a sentence encoding approach where we keep only one output capsule to represent the sentence encoding. The attention weights are dynamically and iteratively refined so that the sentence representation capsule is also refined further.

4.4.2 ***Output Normalization (ON).*** Output normalization with Softmax can be formally depicted as:

$$a_{j|i} = \frac{exp(b_{j|i})}{\sum_j exp(b_{j|i})}$$

This form of normalization is the standard one used by capsule networks. Intuitively, here, the output capsules compete for the attention of the input capsules. The same input capsule cannot be assigned very high (near 1) normalized weights for all the output capsules, but multiple input capsules can be assigned very high normalized weights (near 1) for any given output capsule. In this formulation, we cannot naturally take a sentence encoding approach by using a single output capsule. If there is a single output capsule, there is no competition among the output capsules. Thus, $a_{j|i}$ will be always 1 regardless of $b_{j|i}$. Therefore, there is no meaning in refining or iterative routing. We can still, nevertheless, create a sentence encoding representation by first having multiple output capsules and then concatenating them but we do not explore that direction in this work.

5 DATASET

We run our models on two different text classification datasets: Reuters-21578, arXiv Abstract Paper dataset (AAPD). Both are multi-label classification datasets. Reuters has 90 classes and AAPD has 54 classes.

6 HYPERPARAMETER TUNING

We initially used Bayesian Optimization with Hyperopt [2] to tune the hyperparameters but we found that the resulting hyperparameters for the routing algorithms were worse than intuitively chosen ones. Instead, we use the hyperopt-optimized hyperameters for the CNN baselines, and extended upon those hyperparameters for every other models. We kept similar dimensions and structure for every model. We also manually run a different experiments with different hyperparameters for the routing algorithm but the algorithms did not seem too sensitive to the hyperparameters in the range that we manually tried. The exact hyperparameters will be available in the supplementary file (the submitted project implementation).

7 EVALUATION

We run each of our models five times (with different random seeds) and report the mean accuracy, mean micro F1, along with their standard deviations, and the maximum micro-F1 (among the five runs). Even though we show the accuracy, it is not a very reliable metric as the classes are imbalanced. Since some classes can have very few samples, macro-F1 can get much lower just because a few samples (which happens to be the majority of the samples of a certain class) get misclassified. So we choose micro-F1 instead. We consider micro-F1 here as the more reliable metric striking a balance between the harshness of macro-F1 and the leniency of accuracy.

8 RESULTS

In the results, we see that none of the routing algorithms are actually able to outperform the CNN baselines. The result is not actually all too surprising if we closely analyze the results in Yang et al. [24].

Model	AAPD			Reuters		
	Accuracy	micro-F_1	max micro-F_1	Accuracy	micro-F_1	max micro-F_1
CNN	97.16% ±0.084	0.691 ± 0.004	0.698	99.58% ±0.008	0.838 ±0.003	0.842
$CNN_{+attention}$	97.21% ±0.083	0.693 ±0.002	0.695	99.51% ±0.012	0.815 ±0.004	0.820
Dynamic Routing	96.82% ±0.049	0.624 ±0.003	0.629	99.35% ±0.026	0.738 ±0.008	0.749
Heinsen Routing	96.82% ±0.012	0.646 ±0.002	0.650	99.28 ±0.040	0.723 +- 0.012	0.746
DSA	95.98% ±0.441	0.545 ±0.092	0.644	98.14% ±0.401	0.421 ±0.054	0.490
DSA Sentence Encoding	98.15 ±0.485	0.421 ±0.060	0.473	98.15 ±0.485	0.421 ±0.060	0.503
No-Routing	96.81% ±0.328	0.664 ±0.019	0.684	99.13 ±0.098	0.702 ±0.232	0.731
CoDA Routing ON	96.63% ±0.320	0.657 ±0.023	0.678	99.28% ±0.065	0.733 ±0.021	0.761
CoDA Routing IN	96.96% ±0.099	0.675 ±0.002	0.678	99.14 ±0.143	0.680 ±0.045	0.732
CoDA Routing IN Sentence Encoding	96.34% ±0.09	0.633 ±0.005	0.637	99.12 ±0.076	0.666 ±0.028	0.706
CoDA Routing IN - Highway	96.19 ±0.428	0.633 ±0.025	0.651	99.25 ±0.107	0.713 ±0.038	0.751

Table 1: Performance on AAPD and Reuters Dataset

Yang et al. [24] do manage to outperform the CNN baseline in the datasets that they use, but they also use the expensive convolutional capsule layer which we do not. Interestingly, another variant of their architecture with a slight difference also fails to outperform the simpler CNN baseline despite still having the expensive convolutional capsule layer. Heinsen [10] do seem to achieve decent performance on text classification with their routing algorithm, but in contrast to other methods that they compare against, they use GPT-2 embeddings [18] and they utilize capsule routing over all of its layers. It's not, therefore, clear if the improvement is due to the routing or the use of all layers of GPT-2. They do not compare against any baselines that use GPT-2 with simple layer aggregation and attention mechanism based strategies for classification. Yoon et al. [25] achieved high performance on Natural Language Inference with DSA but they also use it on a high parameter setting with a larger dataset (around 10 times more than AAPD). Regardless, in the settings of our comparisons, all the routing algorithms significantly underperform compared to simpler CNN and CNN-attention based baselines. Among the routing algorithms themselves, the sentence-encoding based methods seem to perform comparatively poorly. The non-routing method is competitive, especially in AAPD. Our new CoDA Routing based methods perform better than most in AAPD dataset, but not as good compared to other algorithms in Reuters dataset. It can be possible that our algorithm needs more data to train better (Reuters have much less sample than AAPD)/ Still one variant of CoDA Routing perform better than most others even in Reuters.

9 CONCLUSION

We evaluated some of the main routing algorithms that have been used for NLP in comparable settings. We conducted an ablation test of removing the routing mechanism. We experimented with two new modifications to existing routing algorithms and we conducted theoretical and empirical analysis on the different types of normalization and their connection to attention and sentence encoding.

10 FUTURE WORKS

Future Works may include exploring the routing algorithms in a different setting. Possibly similar to Heinsen [10], we can explore using routing algorithm to elegantly aggregate different layers and temporal positions of hidden states from one of the big pre-trained transformer language models and route them to class representations. It would be also good to better experiment with multiple (at-least two) layers of capsule layers. Scalability and adaptive routing iterations [28] are also interesting future directions to consider.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:cs.CL/1409.0473

[2] J. Bergstra, D. Yamins, and D. D. Cox. 2013. Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. *To appear in Proc. of the 30th International Conference on Machine Learning (ICML 2013)* (2013).

[3] Zhenhua Chen, Xiwen Li, Chuhua Wang, and David Crandall. 2019. P-CapsNets: a General Form of Convolutional Neural Networks. *arXiv preprint arXiv:1912.08367* (2019).

[4] Zhen Cheng, Zaixiang Zheng, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2019. Multi-Perspective Inferrer: Reasoning Sentences Relationship from Holistic Perspective. *arXiv preprint arXiv:1911.03668* (2019).

[5] Zi-Yi Dou, Zhaopeng Tu, Xing Wang, Longyue Wang, Shuming Shi, and Tong Zhang. 2019. Dynamic layer aggregation for neural machine translation with routing-by-agreement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 86–93.

[6] Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, Jianxin Liao, Chun Wang, and Bing Ma. 2019. Investigating Capsule Network and Semantic Feature on Hyperplanes for Text Classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 456–465. https://doi.org/10.18653/v1/D19-1043

[7] Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, Jianxin Liao, Tong Xu, and Ming Liu. 2019. Capsule Network with Interactive Attention for Aspect-Level Sentiment

Classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 5489–5498. https://doi.org/10.18653/v1/D19-1551

[8] Yufei Feng, Xiaodan Zhu, Yifeng Li, Yuping Ruan, and Michael Greenspan. 2018. Learning Capsule Networks with Images and Text. (2018).

[9] Shuhao Gu and Yang Feng. 2019. Improving Multi-head Attention with Capsule Networks. In *Natural Language Processing and Chinese Computing*, Jie Tang, Min-Yen Kan, Dongyan Zhao, Sujian Li, and Hongying Zan (Eds.). Springer International Publishing, Cham, 314–326.

[10] Franz A Heinsen. 2019. An Algorithm for Routing Capsules in All Domains. *arXiv preprint arXiv:1911.00792* (2019).

[11] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian Error Linear Units (GELUs). arXiv:cs.LG/1606.08415

[12] Geoffrey E. Hinton. 1990. Mapping part-whole hierarchies into connectionist networks. *Artificial Intelligence* 46, 1 (1990), 47 – 75. https://doi.org/10.1016/0004-3702(90)90004-J

[13] Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. 2011. Transforming Auto-Encoders. In *Artificial Neural Networks and Machine Learning – ICANN 2011*, Timo Honkela, Włodzisław Duch, Mark Girolami, and Samuel Kaski (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 44–51.

[14] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. 2018. Matrix capsules with EM routing. In *International Conference on Learning Representations*. https://openreview.net/forum?id=HJWLfGWRb

[15] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1746–1751. https://doi.org/10.3115/v1/D14-1181

[16] Jian Li, Baosong Yang, Zi-Yi Dou, Xing Wang, Michael R. Lyu, and Zhaopeng Tu. 2019. Information Aggregation for Multi-Head Attention with Routing-by-Agreement. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 3566–3575. https://doi.org/10.18653/v1/N19-1359

[17] Inyoung Paik, Taeyeong Kwak, and Injung Kim. 2019. Capsule Networks Need an Improved Routing Algorithm. In *Proceedings of The Eleventh Asian Conference on Machine Learning (Proceedings of Machine Learning Research)*, Wee Sun Lee and Taiji Suzuki (Eds.), Vol. 101. PMLR, Nagoya, Japan, 489–502. http://proceedings.mlr.press/v101/paik19a.html

[18] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. (2019).

[19] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic Routing Between Capsules. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 3856–3866. http://papers.nips.cc/paper/6975-dynamic-routing-between-capsules.pdf

[20] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway Networks. arXiv:cs.LG/1505.00387

[21] Yi Tay, Anh Tuan Luu, Aston Zhang, Shuohang Wang, and Siu Cheung Hui. 2019. Compositional De-Attention Networks. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 6132–6142. http://papers.nips.cc/paper/8845-compositional-de-attention-networks.pdf

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[23] Liqiang Xiao, Honglun Zhang, Wenqing Chen, Yongkun Wang, and Yaohui Jin. 2018. MCapsNet: Capsule Network for Text with Multi-Task Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 4565–4574. https://doi.org/10.18653/v1/D18-1486

[24] Min Yang, Wei Zhao, Jianbo Ye, Zeyang Lei, Zhou Zhao, and Soufei Zhang. 2018. Investigating Capsule Networks with Dynamic Routing for Text Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 3110–3119. https://doi.org/10.18653/v1/D18-1350

[25] Deunsol Yoon, Dongbok Lee, and SangKeun Lee. 2018. Dynamic self-attention: Computing attention over words dynamically for sentence embedding. *arXiv preprint arXiv:1808.07383* (2018).

[26] Ningyu Zhang, Shumin Deng, Zhanlin Sun, Xi Chen, Wei Zhang, and Huajun Chen. 2018. Attention-based capsule networks with dynamic routing for relation extraction. *arXiv preprint arXiv:1812.11321* (2018).

[27] Xinsong Zhang, Pengshuai Li, Weijia Jia, and Hai Zhao. 2019. Multi-labeled relation extraction with attentive capsule network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 7484–7491.

[28] Wei Zhao, Haiyun Peng, Steffen Eger, Erik Cambria, and Min Yang. 2019. Towards Scalable and Reliable Capsule Networks for Challenging NLP Applications. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 1549–1559. https://doi.org/10.18653/v1/P19-1150

[29] Wanshan Zheng, Zibin Zheng, Hai Wan, and Chuan Chen. 2019. Dynamically Route Hierarchical Structure Representation to Attentive Capsule for Text Classification. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 5464–5470. https://doi.org/10.24963/ijcai.2019/759