On Randomized Classification Layers and Their Implications in Natural Language Generation

Gal-Lev Shalev
Bar-Ilan University, Israel
gallev898@gmail.com

Gabi Shalev
Bar-Ilan University, Israel
shalev.gabi@gmail.com

Joseph Keshet
Bar-Ilan University, Israel
jkeshet@cs.biu.ac.il

Abstract

In natural language generation tasks, a neural language model is used for generating a sequence of words forming a sentence. The topmost weight matrix of the language model, known as the classification layer, can be viewed as a set of vectors, each representing a target word from the target dictionary. The target word vectors, along with the rest of the model parameters, are learned and updated during training.

In this paper, we analyze the properties encoded in the target vectors and question the necessity of learning these vectors. We suggest to randomly draw the target vectors and set them as fixed so that no weights updates are being made during training. We show that by excluding the vectors from the optimization, the number of parameters drastically decreases with a marginal effect on the performance. We demonstrate the effectiveness of our method in image-captioning and machine-translation.

1 Introduction

Deep neural networks enabled breakthroughs in natural language generation tasks such as machine-translation (Zhang and Zong, 2015), image captioning (Hossain et al., 2019), and more. Generating the text is done by employing a conditional language model as the decoder component, responsible for predicting the next word at each step during decoding, as depicted in Fig-1. For predicting the next word, the language model first encodes into a vector $f \in \mathbb{R}^d$, denoted as *context representation*, both the previously predicted words, and the task's related input (such as source sentence in machine-translation or input image in image-captioning). Then, at the classification layer, the context representation is projected onto a set of weight vectors, resulting in a vector termed as *logits vector*. Afterward, a softmax function is applied to output a distribution over the target vocabulary words. The

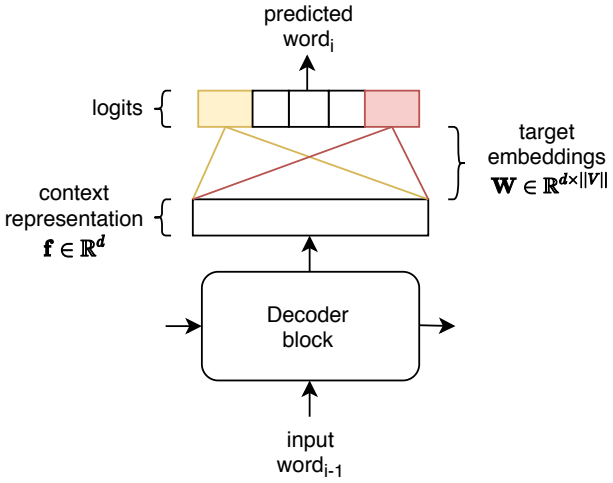

Figure 1: Scheme of a decoding step for predicting the i-th word. The colored boxes and edges represent the logits of two different target words and their corresponding embedding vectors, respectively.

set of weight vectors used for producing the logits, denoted here as matrix $W \in \mathbb{R}^{d \times |V|}$, where V is the target vocabulary. Matrix W can be viewed as $|V|$ vectors, where each is a d-dimensional vector representing a specific word from the target vocabulary, we term these vectors through the paper as *target word embeddings* and *target vectors*, interchangeably. The target embeddings, along with the rest of the model parameters, are estimated so as to minimize the loss function.

During training, an additional set of $|V|$ vectors are being learned to represent the previously predicted words when given to the decoding step. These vectors are referred as *input word embeddings*. Learning the input embeddings during training was shown to improve the performance of NLP classifers and allows achieving a level of generalization that is not possible with classical n-gram language models (Mikolov et al., 2013). Their main advantages are capturing the relationship between words and allowing similar words to have embedding vectors close in space. While a large amount

Proceedings of the Third Workshop on Multimodal Artificial Intelligence, pages 6–11

of work has done to understand the properties and demonstrate the effectiveness of learning the input word embeddings, the benefits of learning the target word embeddings remain unexplored.

In this paper, we show that by randomly drawing the target word embeddings and excluding them from training, the number of trained parameters are drastically decreased with a marginal effect on the performance. By this, we show that the properties captured in the target word embeddings, such as word frequencies and relationships, are surprisingly redundant and may be ignored in low resource environments.

2 Background, Notations and Definitions

Consider a NLG task, such as machine-translation or image-captioning, where for a given input instance x_i, a corresponding sentence S_i should be generated. Typically, an encoder-decoder based neural network is trained for solving the task. The encoder is responsible for encoding the input instance into a vector, while the decoder responsible for generating the next word given the input vector and previously generated words. Commonly, an attention mechanism is incorporated during the decoding (Bahdanau et al., 2014; Xu et al., 2015).

Denote by $D_{train} = \{(x_i, S_i)\}_{i=1}^{N}$ a training dataset where x_i is the input to the model, S_i is the corresponding sentence, and N is the number of training examples. Training the model is done by maximizing the following objective function:

$$\underset{w_1,...,w_{|V|},\theta}{\operatorname{argmax}} \sum_{(x_i,S_i)\in D_{train}} \log P(S_i|x_i) \ ,$$

where θ and $w_1, ..., w_{|V|}$ are the learnable parameters of the model. Since S_i composed of a sequence of words $s_{i1}, ..., s_{ij}$, where j is the length of S_i, a chain rule is applied to model the joint probability over the sentence words as follows:

$$\log P(S_i|x_i) = \sum_{z=1}^{j} \log P(s_{iz}|x_i, s_{i1}, ..., s_{iz-1}) \ .$$

For generating the next word, a context vector $f_\theta \in \mathbb{R}^d$ is learned and is responsible for encoding the given input along with the previously predicted words. Then, the context vector is projected onto each of the target word vectors $w_j \in \mathbb{R}^d$ where $j = 1, ..., |V|$, by calculating the dot-product between the vectors. Afterward, a bias term $b \in \mathbb{R}^{|V|}$ is added, and a softmax function is applied, resulting in a distribution over the target words. Formally:

$$P(s_{iz}|x_i, s_{i1}, ..., s_{iz-1}) = softmax(W \cdot f_\theta + b)$$

Since $w_j \cdot f_\theta = ||w_j|| \cdot ||f_\theta|| \cdot \cos(\alpha_{w_j,f_\theta})$, where α_{w_j,f_θ} is the angle between the vectors, the predicted probability of the word s_j can be written as:

$$\frac{e^{||w_j||\cdot||f_\theta||\cdot\cos(\alpha_{w_j,f_\theta})+b_j}}{\sum_{m=1}^{|V|} e^{||w_m||\cdot||f_\theta||\cdot\cos(\alpha_{w_m,f_\theta})+b_m}} \quad (1)$$

Notice that both the angles and magnitudes of the target word vectors are influencing the predicted probability in Eq-1. The cosine of the angle between w_j and f_θ measures how well the word s_j fits into the context. Hence, interchangeable words have their corresponding target vectors directed at the same angle. The magnitudes of the target vectors control on the predicted probability, in a way that they have a stronger effect on words whose embeddings have direction similar to f_θ, and less effect or even a negative effect on words in other directions. Consider a case where the cosine of the angle between w_j and f_θ is close to 1, meaning that the angle between them is close to 0. In this case, increasing the magnitude $||w_j||$ would result in an increased probability for the word s_j. However, when w_j is directed in an opposite direction to f, the cosine of the angle between them would be close to -1, and therefore, increasing the magnitude would result in a lower probability. By fixing the target vectors and the bias term, the model can maximize the probability in Eq-1 **only by optimizing the vector** f_θ.

In recent work, Press and Wolf (2016) proposed tying the target and input embeddings by using the same vectors to represent both. The paper showed that the performance of weight tied models are on par with learning two separate vectors in machine-translation. However, the method forces the target vectors to have the same dimension as the input embeddings and adds additional computational costs. More recently, several works (Shalev et al., 2020; Hoffer et al., 2018; Shalev et al., 2018) explored the effects of fixing the classification layer in image classification models and demonstrated that the accuracy, number of parameters and out-of-distribution detection ability improve.

In this paper, we empirically show that randomly drawn, fixed target word embeddings allow models to achieve high performance in natural language generation tasks. From an efficiency perspective,

Model	B4	B3	VU	R-L	ME
Tell	27.28	36.59	721	49.72	23.09
Tell-Tied	27.11	36.23	699	49.21	22.83
Tell-Fixed	27.01	36.08	1378	49.34	22.99
Attend	29.12	38.51	784	50.91	24.02
Attend-Tied	29.01	38.11	773	50.76	23.90
Attend-Fixed	28.75	37.93	1026	50.96	23.91

Table 1: Image-captioning evaluation results. B3-4 refer to BLEU3-4. R-L refers to ROUGE-L. ME refers to METEOR. VU refers to vocabulary usage.

fixing these vectors decrease the number of parameters in the classification layer by $d \cdot |V|$. Since the target vocabulary typically contains thousands of words, and the dimension of the context vector f_θ is in hundreds or thousands, the reduction in parameters is significant.

3 Experiments

In this section, we present our experimental results. We evaluate our approach on image-captioning and machine-translation tasks. We start by describing the experimental setup; then, we present the results and analyze the target embeddings.

3.1 Experimental Setup

Image-captioning: For evaluating our approach in image-captioning, we implemented LSTM-based sentence generator as described in (Vinyals et al., 2015), denoted as *tell*. We also implemented an attention-based model as described in (Xu et al., 2015), denoted as *attend*. Additionally, we created identical models where the target embedding vectors are tied (*tell-tied* and *attend-tied*) and also the same models with randomly drawn target vectors without being updated during training (*tell-fixed* and *attend-fixed*). For the fixed target embeddings models, we randomly draw per each cell in the vectors a number in the range of [-10,10]. We evaluated the models on MSCOCO dataset (Lin et al., 2014) and used the standard, publicly available splits, as in previous work (Karpathy and Fei-Fei, 2015). For all models, we set a pre-trained Resnet-101 (He et al., 2016) as the image encoder, provided by the TorchVision package. Due to space limitations, we describe the training procedure in the appendix.

Machine-translation: For evaluating our approach in machine-translation, we used MultiK30 (Elliott et al., 2016), IWSLT 2014 (Cettolo et al., 2014) and WMT-14 datasets. For MultiK30 and IWSLT 2014 sets, we trained an attention-based encoder-decoder model, as described in (Bahdanau et al., 2014). We evaluated the models on English-German and German-English translations. For the WMT-14 EN-DE set we trained a convolutional sequence to sequence model as described in (Gehring et al., 2017). Additionally, we trained the same models with randomly drawn, fixed target vectors, and also with tied-embeddings. The training procedure described in detail in the appendix. We found that translation models with fixed target vectors perform best when the magnitude of the vectors is small, thus we normalized the vectors by dividing them with their L_2 norm.

Dataset	Translation	Non-Fixed	Tied	Fixed
Multi30k	DE-EN	33.02	33.08	33.17
	EN-DE	31.63	31.49	32.12
IWSLT 2014	DE-EN	28.77	28.94	29.03
	EN-DE	25.48	25.66	25.97
WMT-14	EN-DE	25.84	25.62	25.49

Table 2: BLEU4 results for machine-translation.

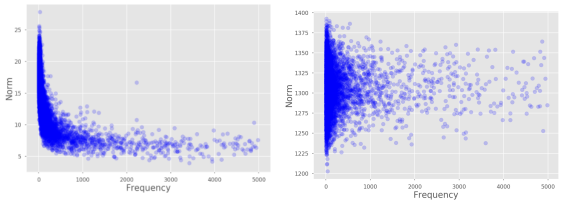

Figure 2: Left is the magnitude of the target vectors learned by *tell* model versus the corresponding word frequencies. Right is the magnitude of the randomly drawn vectors of the fixed *tell* model versus the word frequencies.

3.2 Results

The results for image-captioning and machine-translation are shown in Table-1 and Table-2, respectively. Results suggest that our method of randomly drawing the target embeddings and fixing them during training allows the models to achieve high results in both tasks.

Next, we analyze the learned target word vectors. We find that the word frequencies are reflected in the magnitude of these vectors. As can be seen in Fig-2, target vectors with large magnitude are representing less frequent words. We measured the spearman's rank correlation coefficient between the magnitude of the target vectors ,$\|w_j\|$, and the number of appearances of the corresponding word s_j in the training set. We obtain a strong correlation between the two in all settings. In image-captioning, we obtain a correlation of 0.79 and 0.77 when considering the target vectors of *tell* and

attend, respectively. In machine-translation, for Multi30K DE-EN and EN-DE, we obtain a correlation of 0.84 and 0.93, respectively. For IWSLT DE-EN and EN-DE, we find a correlation of 0.76 and 0.83, respectively. For WMT-14 EN-DE we find a correlation 0f 0.81.

In addition, we observe that the target vectors are able to capture word relationships. Recently, (Press and Wolf, 2016) showed that the target and input vectors of word2vec skip-gram are correlated similarly with human judgments of the strength of relationships between concepts. We followed the same experiment, and found that the target vectors correlate better with the human judgements than the input vectors in 3 out of the 4 tested models. Due to space limitations, results are shown in the appendix.

Interestingly, we noticed that the image-captioning models with fixed target embeddings had an increased vocabulary usage rate (see Table-1) and generated low-frequency words more often compared to the equivalent non-fixed models. In Fig-3, we demonstrate two images for which the non-fixed *tell* model generated the frequent word *bird* (appearing in 5135 training sentences), while the fixed model generated the words *seagull* (appearing in 201 training sentences) and *duck* (appearing in 263 training sentences). More examples can be seen in the appendix. We suspect that the increased usage of low-frequency words might be due to the randomization of the target vectors, which forces **visually similar concepts to have their target vectors far in space**. As a result, the model is encouraged to find a more discriminative representations to distinguish between the concepts. Recall that the cosine-similarity between f_θ and w_j measures how well word s_j fits into the context. If w_j and w_i represent concepts s_j and s_i which are **visually** close but the vectors are far in space, the model would have to find better representations for f_θ to determine whether it should be close in angle to w_j or w_i, as f_θ is the only term that can be optimized in Eq-1 when the target vectors are fixed. In the example above, the non-fixed *tell* model placed the vectors representing the concepts close in space. The cosine-similarity between w_{duck} and w_{bird} is 0.82, and is 0.81 between $w_{seagull}$ and w_{bird}. In contrast, the cosine-similarity between the equivalent target vectors in the fixed models are roughly 0 due to the randomization.

Non-Fixed: a bird that is flying over the water

Fixed: a seagull is standing in the sand on a beach

Non-Fixed: a bird that is standing in the water

Fixed: a duck swims through the body of water

Figure 3: Captions generated by fixed and non-fixed *tell* models.

Model	Non-Fixed	Fixed	%
Tell	18,001,171	13,142,291	27%
Attend	25,342,739	20,483,859	19%
Multi30K DE-EN	13,893,381	10,870,272	22%
Multi30K EN-DE	14,898,861	10,870,272	28%
IWSLT DE-EN	20,237,792	14,307,512	30%
IWSLT EN-DE	21,158,627	14,307,512	33%
WMT-14 EN-DE	36,267,832	28,043,832	21%

Table 3: The number of learnable parameters in each model with the relative decrease percentage.

3.3 Parameters and Computation Efficiency

Recall that the classification layers contains $d \cdot |V|$ parameters, where V is the target vocabulary and d is the dimension of the context vector f_θ. Table-3 demonstrates the significant reduction in the number of learnable parameters. Our method also results in improved computational efficiency compared to the tied-embeddings method (Press and Wolf, 2016). When using tied-embedding, the target vectors are the same as the input vectors, and therefor their dimensions are equal. As a result, the context vector, f_θ, should also be adjusted to have the same dimension as the input and target vectors. In contrast, our proposed method allows to set low dimensional representations, which results in increased computational efficiency at inference. Additionally, the fixed target vectors can be initialized with sparse vectors which can result in memory efficiency. An example is the Hadmard matrix, used by (Hoffer et al., 2018) as the last fully connected layer in image classification models.

3.4 Conclusions and Future Work

In this paper, we demonstrated that by randomly drawing the target embeddings, and setting them as fixed during training, the number of learnable parameters is significantly decreased while allowing to achieve high performance in machine-translation and image-captioning.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, volume 57.

Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. 2017. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2970–2979.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30k: Multilingual english-german image descriptions. *arXiv preprint arXiv:1605.00459*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*, pages 1243–1252. PMLR.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Elad Hoffer, Itay Hubara, and Daniel Soudry. 2018. Fix your classifier: the marginal value of training the last weight layer. *arXiv preprint arXiv:1801.04540*.

MD Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. 2019. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)*, 51(6):1–36.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751.

Ofir Press and Lior Wolf. 2016. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*.

Gil Sadeh, Lior Fritz, Gabi Shalev, and Eduard Oks. 2019. Generating diverse and informative natural language fashion feedback. *arXiv preprint arXiv:1906.06619*.

Gabi Shalev, Yossi Adi, and Joseph Keshet. 2018. Out-of-distribution detection using multiple semantic label representations. *arXiv preprint arXiv:1808.06664*.

Gabi Shalev, Gal-Lev Shalev, and Joseph Keshet. 2020. Redesigning the classification layer by randomizing the class representation vectors. *arXiv preprint arXiv:2011.08704*.

Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele. 2017. Speaking the same language: Matching machine to human captions by adversarial training. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4135–4144.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.

J. Zhang and C. Zong. 2015. Deep neural networks in machine translation: An overview. *IEEE Intelligent Systems*, 30(5):16–25.

4 Appendix

4.1 Training procedures

Image-captioning Training descriptions were preprocessed with basic tokenization, keeping all words that appeared at least 5 times in the training set. Words appearing less are map into *UNK* symbol. For training the models, we use Adam optimizer and set the initial learning rate to 0.0004. We multiply the learning rate by 0.8 for every 8 epochs without improvement in the BLEU score. Training is ended once the model achieves 20 epochs without improvement in the BLEU score. The batch size is set to 32 instances. At training, we apply *teacher-forcing* by feeding at each time step the ground-truth word. During decoding, we use beam-search as the decoding strategy, with a beam size of 10.

Machine-translation Training procedure for MultiK30 dataset is as follows: the training sentences were preprocessed with basic tokenization, keeping all words that appeared at least 2 times in the training set. Words appearing less are map into *UNK* symbol. For training the models, we use Adam optimizer and set the initial learning rate to 0.001. We multiply by 0.8 the learning rate for every 8 epochs without improvement in the BLEU score. Training is ended once the model achieves 20 epochs without improvement in the BLEU score. The batch size is set to 128 instances. At training, we apply *teacher-forcing* by feeding at each time step the ground-truth word.

For training the models on IWSLT we use the same procedure as in MultiK30 with the following modifications: We keep words that appeared at least 5 times in the training set, and filter data to have sentences with max length of 20. The initial learning rate is set to 0.002 and multiplied by 0.25 for every 8 epochs without improvement in the BLEU score. The batch size is set to 64 instances.

4.2 Word Relationships

For evaluating the quality of the non-fixed target vectors in both image-captioning and machine-translation, we follow the evaluation proposed in the *tied embeddings* paper. We calculate the pairwise (cosine) distances between embeddings and correlate these distances with human judgments of the strength of relationships between concepts. Results are shown in Table-4.

Model	Simlex999	MEN	MTurk-771
tell	**0.30**/0.24	0.26/**0.46**	0.20/**0.34**
attend	**0.38**/0.26	**0.49**/0.45	**0.40**/0.32
IWSLT DE-EN	0.07/0.07	**0.16**/0.07	**0.14**/0.11
MultiK30 DE-EN	**0.19**/0.04	**0.36**/0.01	**0.31**/0.06

Table 4: Spearman's correlation between word vectors and human judgments of the strength of relationships between concepts. The correlation of the target vector are in the left column. The correlation of the input vectors are in the right column.

4.3 Diversity

Despite the substantial progress in recent years, sentences produced by existing image captioning methods are still often overly rigid and lacking in variability. Several works (Shetty et al., 2017; Dai et al., 2017; Sadeh et al., 2019) address these issues with an alternative training and inference methods to generate more natural and diverse image descriptions. In Fig-4 we show how simple technique such as fixing the classification layer, which does not require any additional computational cost, might improve the diversity and accuracy of the generated captions.

Figure 4: Examples of captions generated by *tell* and *attend* models. NF and F refer to models with non-fixed and fixed target embeddings, respectively. Low-frequency words are underscored.