Learning to Evaluate Neural Language Models

James O'Neill
University of Liverpool, UK.
Email: James.O-Neill@liverpool.ac.uk

Danushka Bollegala
University of Liverpool, UK
Email: danushka@liverpool.ac.uk

***Abstract*—Evaluating the performance of neural network-based text generators and density estimators is challenging since no one measure perfectly evaluates language quality. Perplexity has been a mainstay metric for neural language models trained by maximizing the conditional log-likelihood. We argue perplexity alone is a naive measure since it does not explicitly take into account the semantic similarity between generated and target sentences. Instead, it relies on measuring the cross-entropy between the targets and predictions on the word-level, while ignoring alternative incorrect predictions that may be semantically similar and globally coherent, thus ignoring quality of neighbouring tokens that may be good candidates. This is particularly important when learning from smaller corpora where co-occurrences are even more sparse. Thus, this paper proposes the use of pretrained model-based evaluation, which assesses semantic and syntactic similarity between predicted sequences and target sequences. We argue that this is an improvement over perplexity which does not distinguish between incorrect predictions that vary in semantic distance to the target words. We find that models that outperform other models on perplexity, do not necessarily perform better on measures that evaluate using semantic similarity.**

***Keywords*-machine learning; language modelling; model-based evaluation**

I. Introduction

Language modelling is an important task in natural language processing and is partly responsible for many of the recent successess of transferable pretrained models that require very little fine-tuning to perform well on a host of various downstream tasks [1], [2], [3], [4]. However, language models are currently evaluated using perplexity, which is directly proportional to the log-likelihood. Although log-likelihood is efficient in that it only evaluates the loss of each token and the corresponding predicted probability for that token, it fails to account for incorrect predictions that may be synonymous with the token, or share some semantic equivalence between predicted sentences and target sentences as a whole. Other conditional language modelling tasks use n-gram overlap measures, such as BLEU in machine translation [5] and CIDEr [6] for image captioning.

We argue that an evaluation measure that takes semantic and syntactic similarities into account results in a measure that reflects human judgements better than perplexity given that model-based evaluators are learned from human annotations of sentence similarity. This also results in generated text that allows for more diverse predictions since it is not restricted to word-level cross-entropy loss.

Contribution: This paper proposes to transfer pretrained models from semantic textual similarity tasks which can be used for evaluating text generation models. Pretraining on semantic textual similarity tasks allows us to optimize for generated sentences that are semantically similar to the ground truth tokens. This becomes particularly important for words that lie in the long-tail of the unigram distribution that have less context to disambiguate from. In such cases, the perplexity is likely to increase given that alternative incorrect predictions are ignored. Concretely, an evaluation measure that accounts for incorrectly predicted alternatives to the target token are is critically important for less frequent tokens. Moreover, we are indirectly leveraging the readily available annotations of semantic similarity between sentences that we argue should be considered for evaluating text generation models.

We also include a simpler non-parametric embedding evaluation measures such as average cosine similarity and Word Mover's Distance (WMD) [7] between the embeddings corresponding to the predicted and target sequences. These model-free embedding similarity measures are relatively fast to compute, and evaluates embedding similarities at the word-level.

II. Related Work

A. Sentence Representations

We briefly introduce recent research on learning universal sentence representations before introducing the corresponding models that can be used for evaluation.

SkipThought: These representations are widely used and have shown good performance for semantic relatedness, paraphrase classification, question classification, sentiment analysis, among many other pairwise sentence tasks, as shown when used as input to simple linear classifiers [8].

QuickThought: Quick-thought vectors [9] reformulate Skipthought training by treating the prediction of the next sentence given the past sentence as a classification problem where the decoder is replaced with a classifier that chooses between a set of candidate sentences. This is less precise than the generative formulation, however it drastically reduces the output dimensionality, leading to faster computation and improved scalability. This is particularly noteworthy in the context work if model-based evaluators are also to be used for optimization of neural language models, in the reinforcement learning setting.

InferSent: Infersent [10] use RNNs with supervision from natural language inference sentence-pairs provided in the Stanford Natural Language Inference (SNLI) [11] dataset. They showed that this *InferSent* model generalized well to other tasks, and in some cases outperforms the unsupervised SkipThought model. We include this in our analysis as to compare how this supervised pretrained model compares to SkipThought in evaluation.

Weighted Meta-Embeddings: Meta-embeddings with a weighted combination of multiple pretrained word-embeddings are used in either an unsupervised way prior to a task, or learned directly for a supervised task on multiple intrinsic and downstream tasks [12]. In the latter case, a reconstruction loss for an autoencoder is used as an auxiliary loss. Similarly, pretrained sentence meta-embeddings can be obtained when used in a Siamese network for the sentence-based pairwise tasks e.g semantic relatedness, paraphrasing etc.

B. Learning The Evaluation Measure

Prior work [13] has argued that current n-gram overlap based evaluation measures are biased and insufficiently correlate with human judgements of the generated response quality. They propose to use an evaluation model that learns to predict the manually annotated scores using a proposed human response dataset. They found that the predictions from the evaluation model had better correlations with human judgements than an n-gram overlap measures such as BLEU.

Novikova et al. [14] have also considered alternative measures for evaluating natural language generation systems. They include a comprehensive evaluation using word-based metrics (tf-idf frequency), n-gram overlap metrics (BLEU, ROUGE, CIDEr), semantic similarity between Latent Semantic Analysis (LSA) word representations with WordNet measures and grammar based metrics (readability, grammaticality). Importantly, they report that metric performance is both data and system-specific, but find that automated metrics reliably work at the system-level. This also reinforces the idea of considering a range of different word-level and sentence-level evaluation metrics, including automated model-based measures.

The difference in our work is that we are proposing pretrained neural network model-based evaluation that not only includes models that are trained on semantic similarity tasks (such as their use of LSA) but other tasks such as paraphrasing and natural language inference, leading to universal sentence representations. We also propose faster model-free alternatives such as WMD and Average Cosine Similarity between word embeddings. Additionally, the model-based scores obtained when computing similarity between sentence representations indirectly takes into account the readily available human scores that were used for supervision, thus, avoiding extra manual annotations.

C. Evaluating Language Models

Chen et al. [15] compare perplexity and word error rate for n-gram models on language modelling and surprisingly find a strong Pearson correlation between both quantities. They suggest that a linear combination of both may lead to a stronger measure. This is one of the first papers that draw attention to alternatives to perplexity and its relation to other measures. Similarly, we analyze perplexity with respect to other measures for neural language models instead of n-gram language models

Marvin and Linzen [16] focus on evaluating syntax in language models by proposing a dataset that assesses the grammaticality of predictions. They pair minimally different grammatical and ungrammatical sentence pairs and train an LSTM language model to analyze whether the LSTM performs better on grammatical sentences as expected. They conclude that although multi-task learning with a syntactic objective can improve performance, there still remains a considerable gap compared to human performance.

Chaganty et al. [17] propose control variates to debias metrics such as BLEU with the help of human judgements, as a tradeoff between improved correlation and labor.

III. Methodology

A. Language Models

Recurrent Neural Network Language Model: For a sequence pair (X, Y) where $X = \{x_1, ..x_t, ..x_T\}$ is the input sequence and $Y = \{y_1, ..y_t, ..y_T\}$ is the target sequence, for $t \in T$ timesteps, x_t is passed to a parametric model along with the previous hidden state vector h_{t-1} to produce an output from its last layer $\tilde{h} = f_\theta(x_t, h_{t-1})$. Here, θ are the parameters of the sequential parametric model (e.g RNN), $h_t = f_\theta(h_{t-1}, y_{t-1})$ is an encoded hidden state vector. The probability $P_\theta(y_t|h_t)$ is then computed using a linear projection, leading to a prediction $\hat{y} = \arg\max \phi(h \cdot W + b)$, where ϕ is a softmax function that normalizes the output to calibrate $P_\theta(y_t|h_t)$, W are the decoder weights and the predicted token is retrieved via the $\arg\max$ operator. The log probability $P(Y|X)$ is maximized as shown in Equation 1, to directly minimize the perplexity in Equation 2.

For unconditional generation (i.e language modelling) $X = \{\langle \text{start} \rangle \cup Y_{1:t-1}\}$, which is the case this paper focuses on. Although, this also applies for conditional generation such as machine translation, question answering and other such sequence-to-sequence problems.

$$\log P(Y_1^t|X) = \sum_{t=1}^{T} \log P(y_t|y_1^{t-1}, X) \\ P(y_t|y_1^{t-1}, X; \theta) = \log P(y_t|h_t; \theta) \tag{1}$$

$$2^{-\frac{1}{T}\log P(x_1,..,x_T)} = \sqrt[T]{\prod_{i=1}^{T} \frac{1}{P(w_i|w_1..w_{i-1})}} \tag{2}$$

The standard way to train the aforementioned RNN language model is to maximize the word-level log likelihood by minimizing the cross entropy loss $\mathcal{L}_{ce}(\theta)$ shown in Equation 3.

$$\mathcal{L}_{ce}(\theta) = -\sum_{t=1}^{T} y_t \log \big(p_\theta(y_t|y_{t-1}, ..y_0)\big) \tag{3}$$

In our experiments, we use a standard 2-hidden layer LSTM sequence model [18] and a 2-hidden layer GRU network [19] with an embedding dimension and hidden state dimension size of $|e| = |h| = 400$. Xavier uniform initialization is used with $(\mu = 0, \sigma = 0.1)$ and gradients are clipped at 0.5 threshold if exceeded at each update, for a batch size $|X_s| = 56$.

Transformer-Based Language Model: We consider the decoder of a multi-layer multi-attention head Transformer language model [3] which uses language model attention heads [3][1] as another evaluation based model. We include this in the analysis for completeness, as the Transformer-based models have recently shown improved performance when compared to recurrent-based models in language modelling and conditional generation tasks such as machine translation [3].

Instead of recurrent connections, Transformers use self-attention over the input context tokens, layer normalization (LayerNorm) and positional embeddings. The output of each attenttion head is concatenated and fed to batch-normalized feedforward layer to generate an output probability distribution for the next target token $t \in T$.

Assume that we have a sequence of vectors $x_1, ..., x_n$ where each vector $x_i \in \mathbb{R}^d$. We define $E \in \mathbb{R}^{n\times d}$ to be a matrix representing the sequence. We define parameters $W_e \in \mathbb{R}^{d\times l}, W_p \in \mathbb{R}^{d\times l}$ and $W_o \in \mathbb{R}^{d\times o}$. Z is defined in Equation 4 where EC is an $n \times o$ matrix of new embeddings. $EW_eW_p^TE^T$ is an $n \times n$ matrix representing the inner products in a new l-dimensional space. $\text{softmax}\big(EW_eW_p^TE^T\big)$ is matrix where each row entry is positive and sums to 1.

$$Z = \text{softmax}\big(EW_eW_p^TE^T\big)EW_o \tag{4}$$

In our experiments we set, $d = 512$, $o = 64$, and the parameters $W_e^j, W_p^j \in \mathbb{R}^{d\times l}$, $W_o^j \in \mathbb{R}^o$ for $j = 1, ..., 8$. Therefore, for the multi-head attention expressed in Equation 5 we use $\mathbb{Z}^j \in \mathbb{R}^{n\times 64}$ and once outputs are concatenated to form $Z^{n\times 512}$.

$$\mathbb{Z}^j = \text{softmax}\big(EW_e^j(W_p^j)^TE^T\big)EW_o^j \\ \mathbb{Z} = \text{concat}(Z^1, ...Z^8) \\ \mathbb{Z}' = \text{Feedforward}(\text{LayerNorm}(Z + E)) \tag{5}$$

We use 8-hidden layers with embedding dimension and hidden state dimensions respectively $|e| = |h| = 512$ with 8 attention heads. An adaptive softmax [20] is used to normalize the output to a probability distribution which we denote as $\phi_{as}(\mathbb{Z}')$, splitting based on term frequency into top 1/10 of words $w \in \mathcal{V}$, 1/10 - 4/10 for the second bin and the remaining 4/10 - 1 for the third bin. Lastly, input, hidden layer and attention dropout is set to 0.1.

B. Pretrained Models for Evaluation

InferSent Evaluation: Conneau et al. [10] use the scoring function in Equation 6 between two encoded sentence pairs (h_1, h_2) where $h_1, h_2 \in \mathbb{R}^d$, correspond to the two sentences $(\mathcal{S}_1, \mathcal{S}_2)$. This scoring function showed to be useful for universal representations, not only natural language inference. We also use the same scoring function for the pretrained InferSent model.

$$\phi\big([h_1, h_2, |h_1 - h_2|, h_1 \cdot h_2] \cdot W + b\big) \tag{6}$$

The model is a Bidirectional-GRU (or BiLSTM) with max (or mean) pooling, as in Equation 7 where g represents the pooling function and e; is the embedding corresponding to word x_i.

$$h = g_{\text{max-pool}}\big([\overrightarrow{\text{GRU}}(e_1, .., e_T), \overleftarrow{\text{GRU}}(e_1, .., e_T)]\big) \tag{7}$$

We also use the self-attentive variation in Equation 8, where the max-pooling operation g is replaced with self-attention that produces a weighted average where $g_{\text{avg}}(\cdot)$ sum the weights to 1 $\forall t \in T$. Hence, attention focuses on the hidden states of important tokens prior to using the scoring function.

$$h = \sum_{t=1}^{T} g_{\text{avg}}(\tanh(Wh_t + b))h_t \tag{8}$$

SkipThought: The original Skipthought [8] paper includes a comprehensive evaluation of CNN-RNN, RNN-RNN and LSTM-LSTM and GRU-GRU encoder-decoder architectures. In our evaluation, we use a Bidirectional GRU encoder decoder architecture which is called bi-skip. SkipThought vectors provide a good unsupervised baseline for model-based text generation evaluation, as they have shown competitive performance against supervised models on the aforementioned tasks of semantic relatendess and paraphrase detection [8].

[1]we follow the hugging face implementation available here: https://github.com/huggingface/pytorch-openai-transformer-lm

Model		↑		↑			↑		↓
	BLEU2	BLEU3	BLEU4	Cosine	WMD	InferSent	SkipThought	Transformer	Perplexity
GRU	24.12/25.98	11.54/12.76	7.71/7.83	69.63/68.20	78.31/79.53	80.52/82.59	87.10/91.23	83.02/81.48	91.45/86.12
GRU-SS	26.40/26.11	12.79/13.14	8.56/9.51	72.59/74.19	80.93/80.16	88.48/90.10	92.14/93.72	85.09/82.02	87.21/82.08
LSTM	26.09/26.20	12.56/12.24	8.34/8.91	68.91/70.08	81.04/80.23	82.01/84.91	84.02/84.33	82.77/82.03	89.32/83.27
LSTM-SS	27.90/28.63	13.15/13.87	8.80/9.04	73.59/72.48	81.43/82.78	84.42/85.06	90.28/91.70	86.10/86.24	84.71/81.11
Transformer	31.49/30.29	14.08/14.82	9.12/9.35	72.09/71.16	79.20/76.25	90.63/91.52	88.02/89.50	84.10/84.55	82.07/79.83

Table I: Language Modelling Evaluation on Penn Treebank (Validation/Test Scores scaled [0, 100]), omitting perplexity

Model		↑		↑			↑		↓
	BLEU2	BLEU3	BLEU4	Cosine	WMD	InferSent	SkipThought	Transformer	Perplexity
GRU	16.24/16.63	7.02/7.71	4.09/4.68	65.30/63.08	74.95/75.49	76.91/78.02	81.27/79.43	80.22/82.76	148.62/132.90
GRU-SS	16.90/17.07	7.56/7.98	4.35/4.85	67.01/65.48	75.22/75.86	83.78/85.93	84.11/86.03	81.27/79.43	141.31/127.80
LSTM	16.58/16.84	7.31/7.76	4.09/4.68	64.15/61.59	76.71/78.03	80.02/76.37	81.90/79.68	81.26/81.52	150.02/137.11
LSTM-SS	17.82/17.49	8.18/8.02	4.09/4.68	69.42/71.09	77.22/80.11	82.01/83.17	83.20/85.41	83.15/85.79	141.66/135.19
Transformer	17.49/17.83	7.25/7.70	3.59/3.85	69.24/72.47	76.95/78.20	82.12/88.28	80.43/84.69	87.35/85.71	120.31/117.98

Table II: Language Modelling Evaluation on WikiText-2 using Model-Based (InferSent & SkipThought) and Model-Free Evaluation Measures

Transformer Evaluation: We also use the original Transformer model [21] for a non-recurrent neural network model-based evaluation, which can also be compared to SkipThought, an RNN-based model that is also trained using unsupervised(-self) learning. To compute scores we use the inner product between the hidden representations of the K most upper layers of the decoder, and compute mean over all K dot products, followed by a Sigmoid non-linearity to normalize $\tilde{s} \in [0, 1]$.

$$\tilde{s}(s_1, s_2) = \sigma\left(\frac{1}{N}\sum_{i=1}^{K}\langle h_1^i, h_2^i\rangle\right) \tag{9}$$

C. Model-Free Embedding Evaluation

Word Mover's Distance Sentence Similarity: We also include the (WMD) [7] for measuring semantic similarity between ℓ_2 normalized embeddings associated with predicted and target words. We also include the average cosine similarity between sentences, which can be considered as an approximation to the optimal transport in WMD. Model-free in this sense means that we do not require a pretrained network for evaluation, only word-level vectors corresponding to predictions and the corresponding targets. Word-level embedding similarities offer a faster alternative to model-based sentence-level evaluation, hence we include it for our experiments.

Average Cosine Similarity: Alternatively, a faster method to evaluate $(\hat{Y}, Y)$ is to compute the average cosine similarity which can be achieved by one of two ways. The first is a straightforward similarity between adjacent word pairs $cos(Y_i, \hat{Y}_i) \quad \forall t \in T$ as shown in Equation 10.

$$s(E_Y, E_Y) = \sigma\left(\frac{1}{|Y|}\sum_{t=1}^{T} cos(Y_t, \hat{Y}_t)\right) \tag{10}$$

We also considered decayed k-pairwise cosine similarity where k is a sliding window span that compares embeddings corresponding to n-gram groupings with a decay factor $\gamma \in [0, 1]$ that depends on the distance such that $\gamma_{(i,j)} = d(Y_i, Y_j)/k \ \forall i, j \in T$.

$$\tilde{s}(Y, \hat{Y}) = \frac{1}{Tk}\sum_{t=1}^{T}\sum_{j=i-k}^{t+k}\gamma_{(i,j)} cos(Y_i, \hat{Y}_j)$$
$$s.t, \quad t \leqslant i \leqslant T - k \tag{11}$$

Dealing with Skewed Scores: Since we are optimizing to exactly predict the target tokens in language modelling, the predicted sequences often result in high embedding similarities by the end of training. Therefore, the differences in [0, 1] normalized scores can appear to be minuscule. To address this we allocate exponentially scaled scores $\tilde{y}$, shown in Equation 12, so that smaller changes of highly similar sentences are distinct, yet still bounded in $[0, 1]$, analogous to perplexity in that it exponentiates the cross entropy loss.

$$\tilde{s} = 1 - (1 - s)2^s = 1 + 2^s s - 2^s \tag{12}$$

We also note that for the average cosine similarity between predicted and target embeddings (denoted as Cosine in Table I and Table II), we pass the similarity as input to a logistic unit $\sigma(\cdot)$ in Equation 13 in order for Cosine scores ($s \in [-1, 1]$) to be comparable with others in $[0, 1]$.

$$\tilde{s}(s_1, s_2) = \frac{1}{\big(1 + \exp(-s/\tau + b)\big)} \tag{13}$$

In our experiments $\tau = 0.1$ and the bias is manually set $b = 8$, as shown in Figure 3 e.g $\tilde{s} = 0.8$ corresponds to an average cosine similarity of $s = 0.82$.

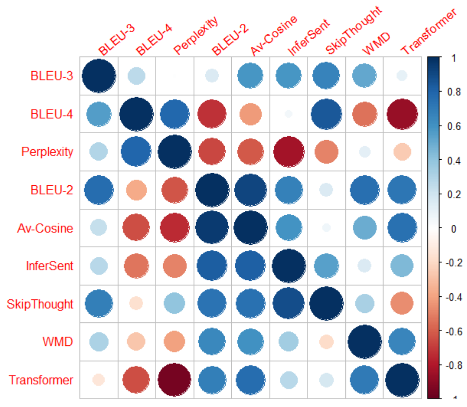

Figure 1: Correlation Matrix of PTB Validation and Test Evaluation Metrics

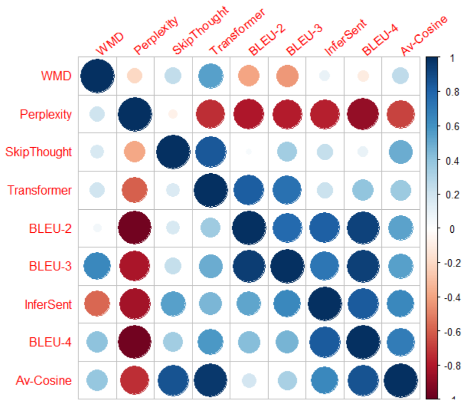

Figure 2: Correlation Matrix of WikiText-2 Validation and Test Evaluation Metrics

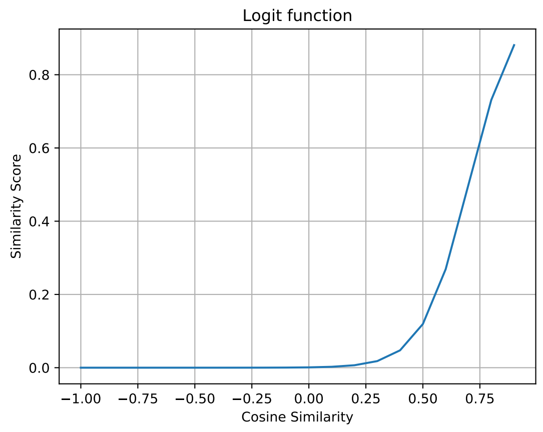

Figure 3: Scaled Similarity Score $s \rightarrow \tilde{s}$ for Average Cosine Similarity (COS) Evaluation

IV. Results

Table I and Table II show the results evaluated on both n-gram overlap based measures, word-level (WMD, Average Cosine Similarity) and sentence-level embedding based evaluation measures (Infersent, SkipThought and Transformer), and lastly perplexity. We consider n-gram overlap to be a strict measure of text generation quality, compared to the embedding-based measures that account for word-level or sentence-level semantic similarity.

The model-based measures that use unsupervised training (SkipThought, transformer) tend to produce high similarity measures between $(\hat{Y}, Y)$ on average, when compared to model-based measures that use supervision (InferSent trained on NLI data).

Interestingly, using scheduled sampling [22] with LSTMs (LSTM-SS) leads to improvements over LSTM without SS, this is similarly found for GRU-SS as well. This is somewhat surprising considering this has been used in the context of language modelling where ground truth tokens are given at test time, unlike text generation, where SS can help mitigate compounding errors. We view SS as inducing noisy samples for language modelling that empirically shows to improve generalization, similar to how multiplicative Gaussian noise on input embeddings achieves superior performance over models without input embedding dropout [23].

We also find that the similarity between $\hat{Y}$ and Y is generally lower across each model for pretrained evaluators that use supervised learning. Concretely, it is more difficult to score well on semantic similarity with InferSent in comparison to SkipThought and Transformer *evaluators*. This can be explained by the fact that the supervision provided by humans in InferSent can be more difficult for a neural language model to learn from scratch without any labels.

From these findings, we posit that standard conditional log-likehood training is limited in accounting for semantically similar sentences since cross-entropy is evaluated at each target token which does not consider good alternative predictions. Moreover, averaging word-level losses does ignores the global coherence of each sentence or paragraph.

Lastly, we note that scaling functions allow for useful comparisons across each measure and lead to better separation between relatively low performing and high performing models on sentence-level semantic similarity metrics, such as the two presented in Equation 12 and Equation 13. Perplexity itself can be considered an exponential scaling of CE loss and is often used to compare language models, as small CE loss alone make it difficult to assess the performance differences between each model.

V. Conclusion

This paper proposed to reconsider how we evaluate neural language models by advocating the use of sentence-based similarity measures between generated sequences and predicted sequences using pretrained pairwise learned models. These models can be trained on supervised tasks such as natural language inference and semantic textual similarity (InferSent) or trained in an unsuperised fashion (SkipThought and Transformer). We argue that this approach can be used in conjunction with perplexity and word-overlap measures, or as an alternative for evaluating text generation systems, along with other model-free embedding similarity measures used and defacto metrics such as perplexity and BLEU. We conclude that neural language models should be evaluated with a variety of different metrics since there is not a clear correlation between them.

References

[1] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

[4] M. E. Peters, W. Ammar, C. Bhagavatula, and R. Power, "Semi-supervised sequence tagging with bidirectional language models," *arXiv preprint arXiv:1705.00108*, 2017.

[5] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.

[6] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4566–4575.

[7] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *International Conference on Machine Learning*, 2015, pp. 957–966.

[8] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in neural information processing systems*, 2015, pp. 3294–3302.

[9] L. Logeswaran and H. Lee, "An efficient framework for learning sentence representations," *arXiv preprint arXiv:1803.02893*, 2018.

[10] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, September 2017, pp. 670–680. [Online]. Available: https://www.aclweb.org/anthology/D17-1070

[11] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," *arXiv preprint arXiv:1508.05326*, 2015.

[12] J. O. Neill and D. Bollegala, "Semi-supervised multi-task word embeddings," *arXiv preprint arXiv:1809.05886*, 2018.

[13] R. Lowe, M. Noseworthy, I. V. Serban, N. Angelard-Gontier, Y. Bengio, and J. Pineau, "Towards an automatic turing test: Learning to evaluate dialogue responses," *arXiv preprint arXiv:1708.07149*, 2017.

[14] J. Novikova, O. Dušek, A. C. Curry, and V. Rieser, "Why we need new evaluation metrics for nlg," *arXiv preprint arXiv:1707.06875*, 2017.

[15] S. F. Chen, D. Beeferman, and R. Rosenfeld, "Evaluation metrics for language models," 1998.

[16] R. Marvin and T. Linzen, "Targeted syntactic evaluation of language models," *arXiv preprint arXiv:1808.09031*, 2018.

[17] A. T. Chaganty, S. Mussman, and P. Liang, "The price of debiasing automatic metrics in natural language evaluation," *arXiv preprint arXiv:1807.02202*, 2018.

[18] M. Sundermeyer, R. Schlüter, and H. Ney, "Lstm neural networks for language modeling," in *Thirteenth annual conference of the international speech communication association*, 2012.

[19] Y. Miyamoto and K. Cho, "Gated word-character recurrent language model," *arXiv preprint arXiv:1606.01700*, 2016.

[20] E. Grave, A. Joulin, M. Cissé, H. Jégou *et al.*, "Efficient softmax approximation for gpus," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1302–1310.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[22] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 1171–1179.

[23] S. Merity, N. S. Keskar, and R. Socher, "Regularizing and optimizing lstm language models," *arXiv preprint arXiv:1708.02182*, 2017.