A Discussion of Semantically Aware Objective Functions for Language Generation Tasks

Jake Bass [*1] **Ishaan Jhaveri** [*1] **Bennett Norman** [*1]

Abstract

This paper summarizes our work from CS 6784 in the Fall of 2017 and our independent study in the Spring of 2018. We previously proposed a set of objective functions that can be used for any text generation task. These functions utilize recent work on high-quality word embeddings to gain an awareness of semantics. In a holistic review of our past and current work, we explore their efficacy and their applicability to the Referring Expression task and Language Generation tasks in general.

1. Introduction

Recent work has shown great success on a myriad of text generation tasks. From image captioning to machine translation, recent advances in deep learning for Natural Language Processing have allowed us to realize impressive performance improvements on many of these tasks. However, relatively little attention has been paid to the objective function used to optimize these networks. In fact, current approaches take a rather naïve approach that: (1) tends to overfit to ground truth sequences, (2) does not take into account that for most language generation tasks, there are many equally plausible correct solutions, and (3) does not take into account semantic information. We previously proposed a set of objective functions to ameliorate each of these issues.

2. Background Information

We previously proposed a set of semantically-aware objective functions and applied them to the Referring Expression task. We will now explain these functions and the Referring Expression task.

[*]Equal contribution [1]Cornell University. Correspondence to: Jake Bass <jab783@cornell.edu>, Ishaan Jhaveri <iaj8@cornell.edu>, Bennett Norman <bdn29@cornell.edu>.

Referring Expression Task: The Referring Expression Task is motivated by the ambiguity of image captioning. The generation task is, given an image and a region of the image containing an object, produce an expression that unambiguously describes the object. The comprehension task is, given an expression and an image, highlight the region of the image that contains the object that the expression describes. This paper discusses the generation task.

Word Mover's Distance: Word Mover's Distance (WMD) (Kusner et al., 2015) is a distance function between text documents. WMD measures dissimilarity between the documents by utilizing semantically meaningful word embeddings and looking at the minimum distance to transform one document, represented as a collection of its word embeddings, into another.

Word Centroid Distance: Word Centroid Distance (WCD) (Kusner et al., 2015) is also a distance function between text documents. Instead of calculating the cost to transform one document to another, WCD computes a centroid for each document, which is the average word embedding for all words in the document weighted by each word's frequency. WCD is defined as the distance between centroids.

Word Embedding Distance: Word Embedding Distance (WED) was created in our previous work. WED takes two same-length documents and computes the distance between the first word of each, the second word of each, etc, and then averages each of these distances.

Gumbel-Softmax: The Gumbel-Softmax trick allows us to use the popular reparameterization trick on a discrete distribution, and thus sample from a discrete distribution while maintaining differentiability (Jang et al., 2016).

3. Our Prior Work

We previously explored the viability of using these metrics as objective functions in our work Semantically Aware Objective Functions for Referring Expression Tasks. We will now summarize that work, and explain how it has informed more recent experimentation.

3.1. Document Distance Metrics as Objective Functions

WMD and WCD were originally formulated as metrics for text document similarity. However, it easy to think of them as objective functions for Referring Expression Generation (REG). While training a model, we simply need to consider the ground truth expression and the generated expression to be "documents". However, doing so raises two major issues: differentiability and positional encoding.

3.1.1. Differentiability

Most models, ours included, do not produce words during training; instead, they produce a categorical distribution over the vocabulary. WMD, WCD, and WED all require words - or more specifically, word embeddings - as input. The simple solution to produce words to be input would be to sample from the predicted categorical distribution, as is done at test time, and then take the sampled words' respective embedding. However, the sampling operation is not differentiable. Thus begs the question: how do we convert a categorical distribution over our vocabulary into a word embedding?

We use two solutions. The first solution is to use the Gumbel-Softmax trick. The second solution is to use what we call an **expected embedding**. For a predicted categorical distribution over a vocabulary, the expected embedding is the weighted average of the embedding of every word in the vocabulary, where each word embedding's weight is the probability of that word in the categorical distribution. Thus, we are able to transform the distribution into an embedding through actions that are all differentiable.

3.1.2. Positional Encoding

The second major issue we encountered was with positional encoding. WMD and WCD have no sense of where words occur in a sentence or document. While this is not an issue for evaluating existing text (e.g. in assessing document similarity), it poses a large challenge when learning to generate text. We came up with two solutions to this problem. The first is to create a hybrid loss that is a weighted combination between Cross Entropy and one of our objective functions. The second solution was to first train a model with Cross Entropy, and then refine it by training more with one of our objective functions. We refer to this condition as training on top of a pretrained model, or pretraining.

3.2. Results of Prior Work

We achieved modest performance improvements over our baseline in many of our experimental conditions. However, there were also many where generated expressions degenerated quickly into nonsense. In particular, models trained with WED or WCD that used either a hybridized loss or pretraining showed at least modest improvements over our baseline in many experimental conditions. However, all models trained with WMD - regardless of whether we used Gumbel vs. Expected Embeddings, or pretraining vs. hybridized loss - produced nonsensical expressions.

3.3. New Direction for This Work

Despite moderate success with our past work, and that there was much more to explored, we decided to move in a new direction. We thought that there must be a smarter, simpler way to incorporate semantic information into an objective function. We sought a method that was: more robust - it could be trained from scratch, didn't require hybridization and the accompanying hyperparameter optimization, and naturally encoded position; simpler - didn't require the trickery of Gumbel or expected embeddings; and more interpretable. In pursuit of these goals, we created the modified Cross Entropy (mCE) objective function. TODO: Finish.

4. Experiments

4.1. Baseline Architecture

In following our new direction, we switched from using our experimental architecture from past work to the current state of the art (SOTA) architecture. At this time, that is a CNN-LSTM architecture from (Yu et al., 2016). They incorporate a Referring Expression Comprehension module and Reinforcement Learning module to the CNN-LSTM.

4.2. Modified Cross Entropy Loss

In our prior work, we extensively cover Cross Entropy loss and its shortcomings. See section 2.3 of our previous paper. For a single prediction, Cross Entropy Loss is defined as:

$$-\sum_{j}^{C} p_j \log q_j \tag{1}$$

where q_j is the predicted probability of the j^{th} token, p_j is the ground truth probability of the j^{th} token, and C is the size of (total number of tokens in) the vocabulary. This value is then averaged across all predictions.

For each ground truth label, there is a ground truth distribution over the vocabulary p. This distribution can be thought of as a vector of length C. We define our ground truth matrix G as a C by C matrix where each row is the ground truth distribution vector for a given ground truth label.

For Cross Entropy, the correct token is assigned probability 1 and every other token is assigned probability 0. Consequently, the ground truth matrix for Cross Entropy is the

identity matrix. For mCE, we seek to set a ground truth matrix that incorporates semantic information when defining each ground truth distribution.

4.3. mCE Formulations

For each of the following formulations, we explain here the intuition behind the formulation and therefore how the mCE matrix was populated to test this intuition. We present the results of the corresponding experiments that tested each of these formulations in the Results section.

1. **Softmax over Pairwise Cosine Similarity Matrix**: The first mCE formulation we tried was born out of our assumption that CE ignores semantic information when it uses 1-hot vectors to represent the ground truth distribution of a given ground truth label. One way of incorporating semantic information in the ground truth distribution for a given ground truth label is to assign a probability to each token in the vocabulary that is a function of the similarity between that token's embedding and the ground truth label token's embedding. So naturally the ground truth label token would still have the highest probability but if the similarity metric truly encodes semantic similarity, synonyms will also have a high probability, and unrelated words will have a low probability. We tried Euclidean Distance and Cosine Similarity as the similarity metric and found empirically that Cosine Similarity works better. So we populated the matrix as follows: we first created a pairwise cosine similarity matrix of size NxN (where N is the vocab size) and then derived probabilities for a given ground truth label y_i, p_j for each token in the vocab, t_j as:

$$p_j = \sigma(-\lambda \cdot cossim(e_{t_j}, e_{y_i})) \tag{2}$$

   Where σ is the softmax function, λ is an empirically set parameter of the softmax function, and e_x is the embedding for a token, x.

2. **K-Nearest Neighbors with Similarity Score Threshold**: Empirically we found that the above softmax approach leaks a lot of probability mass to tokens that have very low similarity scores with the ground truth label token, since there are so many of them. We also found that some tokens have only very few other tokens in the vocabulary they are similar to, or none at all. So we employed two methods of controlling for these empirical observations. In the row that encodes the ground truth distribution for a given ground truth label, we tried only assigning non-zero values to the K tokens whose embeddings had the highest similarity scores to this ground truth label token's embedding. We also made sure the similarity score was above an empirically chosen threshold, else the token was given a probability of 0, even if its embedding was one of the embeddings with the K highest similarity scores to this ground truth label token's embedding. To derive a given token's distribution, we divided the probability mass equally among all the other tokens that were determined to have non-zero probabilities in the above manner.

3. **Synonyms Derived from WordNet Synsets**: Princeton University's WordNet (wor) lexical database contains *synsets* for each word in English. For our purposes the tokens in these synsets can be treated as synonyms for a given token in our vocabulary. Given the KNN shortcomings we discuss in the Results section, we tried a formulation of mCE where, in the row that encodes the ground truth distribution for a given ground truth label, we only assign non-zero values to those tokens in the vocabulary that occur in the ground truth label token's synset. To derive a given token's distribution, we divided the probability mass equally among all the other tokens that were determined to have non-zero probabilities in the above manner. Note this approach marks a departure from probabilities derived in any way from similarities between word embeddings.

4. **Handpicked Synonyms of 200 most common tokens**: As described in the Results section, the synset synonyms suffered from some shortcomings. We decided then to try the simplest possible formulation of probabilities derived from synonyms, with the intention of understanding whether there is any promise at all in this direction. We determined the distribution for a given token as follows. If it was not one of the 200 most common tokens in the training data, its row is just the CE 1-hot vector. If it is one of the 200 most common tokens, we printed out the other tokens in the vocabulary that occur in this token's synset and handpicked those which were assigned a non-zero probability. We made sure not to pick words for preposition words because we discovered from the synset synonyms experiment that the best distribution for such words is just a 1-hot distribution of the word itself.

5. **Handpicked Antonyms of 200 most common tokens**: We tried one other direction. We decided instead of promoting similar words, to demote dissimilar words. As described in the Results section, the synset synonyms suffered from some shortcomings. We decided then to try the simplest possible formulation of probabilities derived from synonyms, with the intention of understanding whether there is any promise at all in this direction. We determined the distribution for a given token as follows. If it was not one

of the 200 most common tokens in the training data, its row is just the CE 1-hot vector. If it is one of the 200 most common tokens, we printed out the other tokens in the vocabulary that occur in this token's WordNet antonym set and handpicked those tokens which were assigned a -1 "probability". We then only assigned a probability of 1 to the token itself.

Figure 1. METEOR and CIDEr scores from our 5 classes of experiments.

Model Description	CIDEr	METEOR
Yu's SOTA	1.064	**0.236**
Handpicked Synonyms	1.029	0.235
Synset Synonyms	0.49	0.203
mCE with partial softmax	**1.085**	**0.236**
KNN with Cosine Similarity	1.056	0.234
Antonyms	0.052	0.014

5. Results

Table 4.3 includes the best scores of our 5 classes of experiments and Yu's SOTA METEOR and CIDEr scores. All reported scores use a beam size of 3 and Yu's rerank mechanism for evaluation.

- Our most successful experiment is mCE with normalized embeddings and partial softmax. It performs as well as Yu's SOTA on METEOR and marginally better on CIDEr. We attribute the small jump in CIDEr to noise because the conditions we used for this experiment resulted in a similarity matrix that is very close to cross entropy and because other mCE experiments produced slightly lower scores then Yu's SOTA.
- The shortcomings of the KNN Experiment are twofold:
  - Some tokens, like "man" and "woman" are assigned very high similarity scores, perhaps because embeddings are derived from the context of a word and "man" and "woman" have similar contexts in the corpora the embeddings were derived from. Therefore the KNN mCE matrix guides the algorithm to allow these tokens to be used in place of each other which is actually bad for the referring expression task, since a picture of a man should not be described as a picture of a woman.
  - Some tokens, especially prepositions and positional words have very specific meanings. Consider the word "bottom". In the training data, this word is used mostly to denote a positional attribute of an object, but its closest neighbor using cosine similarity is "behind" perhaps from its other meaning, meaning a person's "bottom" or "behind". Therefore the KNN mCE matrix guides the algorithm to allow these tokens to be used in place of each other which is actually bad for the referring expression task, since when used to denote a positional attribute of an object, these words have vastly different meanings.
- The handpicked synonym experiment serves as a controlled test to confirm our initial intuitions. However, this tests performs slightly worse than Yu's SOTA.
- The synset synonyms tests performs substantially worse because many words in the vocabulary such as propositions do not have useful synonyms. Using antonyms to populate the similarity matrix clearly does not perform well.

6. Conclusion

Referring Expression Generation requires precise, discriminative language. While incorporating semantic information into an objective function may help a model generalize more effectively and prevent overfitting, sometimes that generality may actually decrease performance. We think that may be the case with our work. In other tasks, word pairs like man and woman, first and second, or baseball and basketball are all very similar. However, for the Referring Expression task, interchanging these words would be disastrous. Consequently, we think that semantically aware objective functions would be much better suited for other domains. We leave this to future work.

We also have concluded that word embeddings, when compared at the individual word-level as opposed to the sentence- or document-level, do not effectively encode semantic meaning. While this area is ripe for future exploration, initial results do not seem promising. However, supervising embeddings and contextual embeddings could ameliorate this issue.

Differentiability and positional encoding remain concerns for objective functions similar to ours. While the Gumbel-Softmax trick did not work with WMD, it has potential to work in other formulations, such as with WED. Hybridization and pretraining with our objective functions are also areas for future work; similarly, expected embeddings have shown promise in tandem with hybridization and pretraining and deserve future exploration.

Lastly, recent work has shown success integrating other non-differentiable objective functions via a Reinforcement Learning policy-gradient approach. We believe WMD and other semantically aware objective functions fit well with this approach.

References

What is wordnet? URL https://wordnet.princeton.edu/.

Jang, Eric, Gu, Shixiang, and Poole, Ben. Categorical reparameterization with gumbel-softmax, 2016.

Kusner, M. J., Sun, Y., Kolkin, N. I., and Weinberger, K. Q. From word embeddings to document distances. In *ICML*, 2015.

Yu, Licheng, Tan, Hao, Bansal, Mohit, and Berg, Tamara L. A joint speaker-listener-reinforcer model for referring expressions. *CoRR*, abs/1612.09542, 2016. URL http://arxiv.org/abs/1612.09542.