Improved Beam Search Diversity for Neural Machine Translation with k-DPP Sampling

Max Spero
Department of Computer Science
Stanford University
maxspero@stanford.edu

Jon Braatz
Department of Computer Science
Stanford University
jfbraatz@stanford.edu

Abstract

Beam search is widely used in natural language generation tasks to evaluate decoder outputs and translate them into comprehensible sentences. However, the usual method of extending partial outputs greedily by choosing the top-k most probable extensions at each time step can lead to the beam getting populated with correlated samples which reduces efficiency and could crowd out other viable outputs. We explore an update to beam search, by sampling using determinantal point processes. This promotes diversity of samples evaluated and mitigates correlations, resulting in a search over a more diverse sample space. We find that a hybrid approach is able to match the BLEU score of top-k approaches while placing a focus on hypothesis diversity during its beam search.

1 Introduction

Beam search has been an important tool for neural machine translation since the first NMT models were published [9]. It is widely used because it is quite efficient to compute, reducing the search space of a machine translation task from $O(|V|^m)$ to $O(|V|k^m)$ (where V is the vocabulary set and m is the maximum length of a sentence). Additionally, beam search can be performed in constant memory (for fixed k), which is not the case in a traditional tree traversal. This is a simple, strong algorithm, however a low k can result in all beams being saturated with highly correlated outputs. Increasing k too much, however, greatly increases the runtime of beam search.

Determinantal point processes (DPPs) are a technique for sampling that aims to sample a diverse subset of points from a set. We hypothesize that using DPPs for sampling can increase the diversity within a fixed beam size for machine translation models. In this paper, we expand on this idea and evaluate it both theoretically and on real-word data.

2 Preliminaries

2.1 Beam Search

State of the art Neural Machine Translation (NMT) systems like Google's seq2seq system [9] and GNMT [11] use neural networks and large corpora of source/target language pairs to learn two complementary components, namely an encoder and a decoder. On inference, the encoder takes as input a sequence of word vectors in the source language and outputs an intermediate state vector. The decoder builds a translation in the target language by successively choosing words from the category to append to the current partial translation. It does this by taking as input the intermediate state vector and a word vector from the target language, and at each time step outputting the next state vector and a probability distribution over words in the target language. At each step of decoding, a word is chosen according to that distribution to be included as the next word in the translation of the source

sentence, and its word vector is fed back as input into the decoder on the next time step. The goal is to output the translation that is most likely, where the probability is given by the product of probabilities of each word from the output distributions at each time step.

Finding the "best" translation according to this metric is thus a tree-search problem that would take exponential time and space to accomplish. The problem is made tractable in practice by using k-wide beam search, which maintains a set of k candidate partial translations and their cumulative probabilities, and chooses the next k candidates by considering extensions of the current candidates by all words in the vocabulary and selecting the k candidates from this set with the highest cumulative probabilities. We consider this form of beam search, which we refer to as "top-k" or "vanilla" throughout our paper, as our baseline to improve upon. Alg. 1 describes vanilla beam search in the same format that we describe our own approaches to this problem.

Algorithm 1 Top-k beam search

Input: Vocabulary V, beam size k

$H_0 \leftarrow \{[< s >]\}$
$H_{complete} \leftarrow \emptyset$
$t \leftarrow 0$
while $|H_{complete}| \neq k$ **do**
    $k_{live} \leftarrow k - |H_{complete}|$
    $H_{temp} \leftarrow$ top_scores$(k_{live}, \{[h, w] | h \in H_t, w \in V\})$
    $H_{complete} \leftarrow \{h | h \in H_{temp}, h_{t+1} =< \backslash s >\}$
    $H_{t+1} \leftarrow \{h | h \in H_{temp}, h_{t+1} \neq< \backslash s >\}$
    $t \leftarrow t + 1$

2.2 Determinantal Point Processes (DPPs)

Each step of k-wide beam search involves choosing a set of k new candidates using the probability distributions output by the decoder. We can think of each set of k candidates as having a certain probability derived from the decoder outputs, and so choosing greedily corresponds to choosing according to the mode of this distribution at each time step. Formally, a probability distribution over the set of subsets of a discrete set $\mathcal{Y}$ is called a point process, and in vanilla beam search the probability of a subset of candidates of size k is simply given by the product of the probabilities of each candidate in the set. Since each candidate affects the total probability of the subset multiplicatively and independently, there are thus no "interaction terms" between candidates in this implicit vanilla beam search distribution.

An example of a point process that does include interaction terms and suppresses the probability of subsets containing "similar" elements is called a determinantal point process. Since in this paper we will explore the effect of changing the vanilla beam search point process to a determinantal point process, we review the terminology associated with them.

2.2.1 DPP Definitions

Formally, $\mathcal{P}$ is called a determinantal point process if, when a random set $\boldsymbol{Y} \subseteq \mathcal{Y}$ is drawn according to $\mathcal{P}$, we have for every $A \subseteq Y$:

$$\mathcal{P}(A \subseteq \boldsymbol{Y}) = \det(K_A)$$

for some positive semidefinite matrix K_A with eigenvalues all less than or equal to 1 indexed by the elements of $\mathcal{Y}$. L-ensembles are a class of DPPs defined using a real, symmetric matrix L indexed by the elements of $\mathcal{Y}$:

$$\mathcal{P}_L(\boldsymbol{Y} = Y) \propto \det(L_Y)$$

Since L is real and symmetric, it can be written as $L = B^T B$ for some matrix B that is in practice often a matrix with feature vectors as columns. A k-DPP is obtained by conditioning a standard DPP on the event that the set $\boldsymbol{Y}$ has cardinality k:

$$\mathcal{P}_L^k(Y) = \frac{\det(L_Y)}{\sum_{|Y'|=k} \det(L_{Y'})}$$

Since k-wide beam search is concerned with subsets of a fixed size k, our main concern will be k-DPPs specifically in what follows.

3 Related Work

DPPs were first broadly introduced to the field of machine learning by Kulesza in [4] for forming diverse sets, with applications to image search, document summarization, and pose estimation. DPPs were used for both inference and training, and achieved the highest performance in their image search task by combining multiple DPP L-ensembles in a mixture model. The same author introduced the k-DPPs and algorithms for sampling them in [3], which we use in our method using code found from (https://github.com/ChengtaoLi/dpp). Efficient sampling for k-determinantal point processes [6] provides further improvements on the runtime bounds over naive implementations of k-DPP sampling.

Song et. al. [8] incorporate DPPs into a neural conversation model based on seq2seq in order to increase diversity of responses to queries. They propose three DPP-based methods for diverse decoding in their conversation system. Their DPP-R algorithm re-ranks the outputs generated by vanilla beam search via DPP sampling, thus incorporating diversity after decoding. Their DPP-D method applies DPPs at every step of the decoding process, much like our method, which incorporates diversity in decoding. Their DPP-D-DivNet applies DPPs at every step of the decoding process and includes a second application of DPPs in what they call a "diversity net", which incorporates diversity in decoding but it specific to conversation models and not applicable to our problem.

Vijayakuman el al. [10] tackled the issue of diverse beam search with applications to image captioning and proposed an approach that they called "Diverse Beam Search". The reasons they cite for attempting to add diversity to beam search include:

- Low diversity of beams leading to computational inefficiency since essentially the same computations are being repeated with no significant gains in performance.
- loss-evaluation mismatch, i.e. improving probabilities by increasing beam width often leads to lower performance on the task since "safer" outputs are chosen over more correct ones.
- tasks with significant ambiguity in acceptable answers, i.e. tasks with multiple correct answers such as in image captioning or translation, are best solved by algorithms that can capture this ambiguity in their outputs.

They did not use DPPs in their solution, and they instead added an extra cost terms to their beam search score expression that penalized having multiple beams with similar n-gram statistics.

4 Approach

Our goal is to replace the greedy step in k-wide beam search that selects the k extensions to the partial translation with a step that would choose a semantically diverse set of k extensions using a k-DPP. Our approach is to create a feature vector for each candidate extension with magnitude equal to the square root of its probability according to the decoder and direction given by the direction of the next hidden state of the decoder for that candidate. Considering these feature vectors as columns in a matrix B, the L matrix in our L-ensemble is given by the Gram matrix B^TB. This defines a probability distribution over subsets of our candidate set of cardinality k weighted by the probability of each candidate in the subset as well as a multiplicative diversity score from the determinant part of the k-DPP calculation that penalizes subsets containing similar candidates. Our output for the beam search step would then be the subset with the highest probability according to this distribution, and we use the probabilities of the chosen extension words to update the running scores for each candidate in the beam search.

However, while sampling from a k-DPP can be done as efficiently as eigendecomposition of the L matrix, finding the mode of the distribution has been shown to be NP-hard by Ko et. al [2] in the context of the closely-related maximum entropy sampling problem. Furthermore, Kulesza et. al. showed in [4] that finding the mode of a k-DPP is NP-hard to approximate within a factor of $8/9 + \epsilon$. For this reason, rather than choosing the most probable subset of k candidates, we sample from the k-DPP distribution using the methodology given in [3]. Furthermore, this sampling process involves an eigendecomposition that is $O(n^3)$ in the number n of dimensions of the matrix. This is the number of hypotheses, in a naive implementation equal to the vocabulary size times number of beams. For runtime considerations, we only sample from $3k$ candidates for the sake of speed.

Algorithm 2 k-DPP beam search

Input: Vocabulary V, beam size k, pool size αk

$H_0 \leftarrow \{[< s >]\}$
$H_{complete} \leftarrow \emptyset$
$t \leftarrow 0$
while $|H_{complete}| \neq k$ **do**
  $k_{live} \leftarrow k - |H_{complete}|$
  $H_{pool} \leftarrow$ top_k$(S, \{[h, w] | h \in H_t, w \in V\})$
  $H_{temp} \leftarrow$ kdpp_sample$(k_{live}, H_{pool}))$
  $H_{complete} \leftarrow \{h | h \in H_{temp}, h_{t+1} =< \backslash s >\}$
  $H_{t+1} \leftarrow \{h | h \in H_{temp}, h_{t+1} \neq< \backslash s >\}$
  $t \leftarrow t + 1$

Algorithm 3 Hybrid beam search

Input: Vocabulary V, DPP-beam size k, pool size αk, top-beam size m

$H_0 \leftarrow \{[< s >]\}$
$H_{complete} \leftarrow \emptyset$
$t \leftarrow 0$
while $|H_{complete}| \neq k$ **do**
  $k_{live} \leftarrow k - |H_{complete}|$
  $H_{temp} \leftarrow H_{kDPP}(V, k, S, H_t) \bigcup H_{topK}(V, m, H_t)$
  $H_{complete} \leftarrow \{h | h \in H_{temp}, h_{t+1} =< \backslash s >\}$
  $H_{t+1} \leftarrow \{h | h \in H_{temp}, h_{t+1} \neq< \backslash s >\}$
  $t \leftarrow t + 1$

We propose two new beam search algorithms. Alg. 2 describes an approach where we use k-DPP sampling instead of top-k for constructing a new beam. The kdpp_sample function is described in theoretical detail in Sec. 5.

Alg. 3 describes a hybrid approach where we combine k-DPP sampling and top-k to construct a beam that is both diverse and contains the most probable hypotheses.

5 Theoretical Justification

Our similarity metric for candidates is the cosine similarity of the decoder hidden units for the partial translations. This is analogous to treating decoder hidden units as a kind of "partial translation embedding", not unlike word embeddings as output by algorithms like word2vec or GloVe. Just as word embeddings capture semantic content insofar as they differ from one hot encodings where each embedding vector is orthogonal to the others, we can gain insight into how our use of the hidden units as embeddings can fail to detect diversity of candidates by considering the behavior of our method when the hidden units for all candidates are orthogonal.

We use the L2-normalized hidden units ϕ_i in our calculation for the L-matrix that we use in k-DPP sampling as:

$$L_{ij} = (\sqrt{\text{score}_i}\phi_i)^T(\sqrt{\text{score}_j}\phi_j) \tag{1}$$

where the scoring function calculates the running probabilities of the partial translations using the equation:

$$\text{score}(y_1, ..., y_t) = p(y_t|X, y_1, ..., y_{t-1}) \cdot \text{score}(y_1, ..., y_{t-1}) \tag{2}$$

with X representing the input sentence from the source language. When the hidden units are orthogonal to each other and $\phi_i^T\phi_j = \delta_{ij}$, the L-matrix becomes diagonal, with the elements on the diagonal given by:

$$L_{ii} = \text{score}(i). \tag{3}$$

In this case, the probability that a subset Y will be sampled by the k-DPP process is given by:

$$\mathcal{P}_L^k(Y) = \frac{\det(L_Y)}{\sum_{|Y'|=k}\det(L_{Y'})} \propto \prod_{i \in Y} \text{score}_i, \tag{4}$$

and so the probability that a subset is chosen is equal to the product of the probabilities of each candidate given the language model. If we could efficiently choose the subset with the highest probability, then this would be equivalent to top-k beam search, as choosing the candidates with the highest probabilities maximizes that product. Choosing the mode of this distribution thus reduces to greedy beam search when the hidden states are orthogonal, and using the k-DPP sampling algorithm to sample from this distribution reduces to choosing the k candidates in vanilla beam search by sampling without replacement.

Orthogonality of candidates is thus a limiting case in which our method reduces to standard beam search that uses the decoder output distribution to sample new candidates instead of choosing them greedily. Strict orthogonality, however, is impossible when the dimension of the hidden states is less than the number of possible candidates, which is the case in practice. A weaker and more realistic degenerate condition than strict orthogonality of hidden states is considering all normalized hidden state vectors to be i.i.d. random vector-valued sampled from the uniform distribution over the unit

sphere. As in the case of orthogonal hidden states, this condition would also correspond to a complete lack of correspondence between cosine similarity and any kind of semantic similarity.

We would like to characterize the mean and standard deviation of the distribution of dot products of two vectors sampled from this distribution. From [1], we know that the pdf, mean μ, and standard deviation σ of the dot product z of two unit vectors sampled randomly from the uniform distribution on the unit n-sphere is given by:

$$p(z) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{n+1}{2})}{\Gamma(\frac{n}{2})} (1 - z^2)^{\frac{n-2}{2}} \qquad \mu = 0 \qquad \sigma = \frac{1}{\sqrt{n+1}} \tag{5}$$

The asymptotic $O(\frac{1}{\sqrt{n}})$ growth of σ with hidden dimension size n is robust to changes in the specific distribution that we're assuming, a result which is ultimately due to the Central Limit Theorem, which says that the sum of n i.i.d random variables (which is the form of that dot product calculation takes) with mean 0 and standard deviation σ approaches a normal distribution with mean 0 and standard deviation $\frac{\sigma}{\sqrt{n}}$ as n approaches infinity. This gives justification to the claim in the first sentence of [1]: "A pair of vectors randomly chosen from a high dimensional Euclidean space are, with high probability, almost orthogonal."

This property of high dimensional Euclidean spaces means that considering hidden states to be random approximates the case of orthogonal hidden states, which we've seen leads our method to reduce to vanilla sampling-based beam search. If hidden state vectors do capture semantic content, then the hope is they will deviate from being randomly distributed and their dot products will not consistently all be approximately 0, which will result in our method sampling more diverse subsets and hopefully lead to higher BLEU scores.

6 Implementation

Our implementation of an NMT system with beam search decoding is based on Google's seq2seq model [9] with attention and sub-word modelling using PyTorch. On each step of decoding, rather than sampling a subset of cardinality k from all $k \cdot |V|$ possible candidates, we introduced a pruning hypermarameter α and sampled from the top αk candidates using the k-DPP sampling algorithms outlined in Algorithm 2 and Algorithm 3, and we used the value $\alpha = 3$.

After examining the $3k$ partial translation candidates with the highest probabilities as given by the decoder softmax outputs, we took one more autoregressive step of the decoder by feeding the chosen extension words back as input to get the next hidden states, normalized each hidden state vector according to L2 norm, scaled by the probability that we obtained from the softmax output from the previous step, formed the Gram matrix of those feature vectors, and performed k-DPP sampling on that matrix using existing code [5] that we modified for our uses.

7 Experiments

7.1 Data

The data we used is a set of English and Spanish translation pairs from TED Talks. We trained a neural machine translation model from assignment 5 and used this single model to evaluate a number of different decoding algorithms based on beam search.

7.2 Evaluation Method

We used BLEU score [7] as our primary evaluation method. BLEU measures the modified n-gram precisions of candidate translations up to a certain n. It is computed with the following equation:

$$BLEU = BP \cdot \exp \sum \frac{1}{N} \log p_n$$

Where p_n is the modified n-gram precision for n and BP is the brevity penalty: $BP = 1$ if $c > r$, $e^{(1-r/c)}$ if $c \leq r$ where c is the length of the candidate translation and r is the length of the reference translation.

BLEU score is a strong signal for how close a translation is to a reference translation of the original sentence. However, it is important to consider that a more diverse beam set, as enabled by DPP-sampling, could result in some synonyms for words being used, which would in turn reduce the n-gram precision and BLEU score of translations.

7.3 Experimental Details

We tested three different beam search algorithms across many values of k, resulting in 37 total evaluations. We evaluated on a test set of 8064 Spanish-English translation pairs that the model has not been trained on. The algorithms tested were:

1. **Top-k beam search.** This is the traditional decoding algorithm used in NMT. Top-k beam search scores all words in the vocabulary for each hypothesis, and takes the k top-scoring hypotheses as the new "beam." Alg. 1 describes this approach in detail. We evaluate this algorithm across k from 1 to 100.
2. **DPP-sampled beam search.** This approach uses determinantal point processes to sample hypotheses for a beam in such a way that prioritizes both probability of each the hypothesis (as predicted by the model), and diversity of the hypothesis set. Alg. 2 describes this approach in detail. We evaluate this algorithm with pool size at a fixed size of 100, and also at a variable (across k) size of $3k$. Both options are evaluated at values of k from 1 to 20.
3. **Hybrid beam search.** This approach is an attempt at gaining the benefits of diverse sampling from DPP-sampled beam search, without forgoing the strong performance of top-k beam search. This approach samples with both DPP and top-k, and then takes the union of the two sets as its hypothesis set. Alg. 3 describes this approach in detail. We evaluate this algorithm with top-m sizes of 1 and 3. Both options are evaluated at DPP sample sizes of k from 1 to 20.

7.4 Results

Fig. 1 describes our results across algorithms and beam sizes. We found that no algorithm incorporating DPP performed strictly better than vanilla top-k beam search in terms of BLEU score. However, qualitative examination of results does show that the negative impacts in BLEU score were often due to synonyms or other creative ways to express the same thought. When evaluated by n-grams for BLEU score, the translations, did not perform as well, despite having a sentence that contained the same meaning.

It is important to note that DPP-sampling alone was not enough to generate good results. Very often, the top word predicted by the model was the same word as the reference translation. Not sampling this word in DPP-sampling would significantly reduce BLEU scores of a translation where this happens. It was this problem that motivated the hybrid beam search approach, which found BLEU scores comparable to top-k beam search, while using DPP sampling to promote improved diversity within beams.

8 Analysis

8.1 Beam size

We did significant analysis on the effect of beam size on neural machine translation. Fig 1 shows a plot of how beam size affects BLEU score for five different beam search algorithms, and we see noticed a few interesting things.

Note the large BLEU score of top-1 beam search, highlighting the importance of taking into account the model's highest-scored output. While choosing the word that the decoder predicts with the highest probability performs quite well for BLEU score, the translations weren't always as good as BLEU score might suggest, due to translations not making perfect grammatical sense at low beam sizes.

Another trend we noticed is that BLEU score tended to rise through beam sizes of 5. After that, they peaked and even started to dip, despite the fact that the model's prediction scores for the chosen translations continued to rise as beam size increased. Beam sizes greater than 1 allow the model more flexibility in choosing grammatically correct multi-word phrases. However, it also seems that

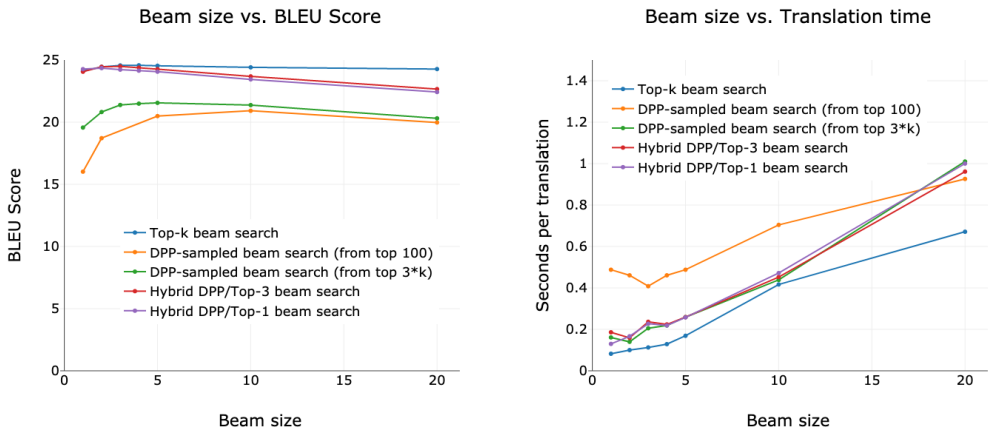

Figure 1: Plots of how different beam search algorithms performed with different beam sizes. (Left) plots beam size against a test set BLEU score. We observe that BLEU score seems to peak around $k = 5$ and decrease from there. (Right) plots beam size against translation time. While DPP and hybrid approaches take longer than top-k beam search, these results show that beam size is the largest factor in determining how long a translation takes. Beam search algorithms are evaluated at beam sizes of $k = \{1, 2, 3, 4, 5, 10, 20\}$.

Beam Size	1	2	3	4	5	10	20	50	100
BLEU Score	24.11,	24.45	24.56	24.57	24.54	24.41	24.27	23.15	15.64

Table 1: BLEU score across beam sizes using top-k beam search. This demonstrates the negative effect on BLEU score that larger beam sizes tend to have, possibly due to overfitting.

having a smaller beam size serves as a guard against overfitting and allows for a model to have greater generalization.

This question of generalization certainly demands a closer look, but Tab. 1 demonstrates this trend for various beam sizes between 1 and 100.

The rightmost graph in Fig. 1 demonstrates how evaluation time increases with greater beam sizes. We see that DPP-sampled search takes a constant factor of time more than top-k search, but does not seem to explode with high k. We see here that beam size is the greatest factor for determining translation time. This implies that an improved beam search sampling algorithm that does not increase the beam size is generally preferable runtime-wise to one that does.

8.2 Qualitative Translation Analysis

Due to the fact that all of these beam search methods used the same underlying NMT model, many of the sentence outputs ended up being the same or dissimilar only in ways that didn't affect translation quality. Here we will compare the outputs of two beam search algorithms.

The first, which we refer to as **Top-k**, is just standard top-k beam search with a beam size of 5.

The second, which we refer to as **DPP**, is our hybrid beam search algorithm which uses a beam of 5 hypotheses sampled by DPP sampling, along with the top 1 hypothesis as scored by the model, which may or may not already be in the first set of hypotheses sampled.

We examine one translation, where the Spanish phrase "Un momento" would translate to "Wait a moment" or "Hold on", but has a literal translation of "A moment"

Source sentence: Y me dije: "Un momento, esto parece interesante."
Reference translation: And I was like, "Hold on. That sounds interesting."
Top-k translation: And I said, "A moment, this seems interesting.
DPP translation: And I said, "Wait a moment, this seems interesting.

Now, examining the the top 5 hypotheses for Top-k at $t = 4$:

```
['And', 'I', 'said', 'to']
```

```
['And', 'I', 'said,', '"A']
['And', 'I', 'said,', '"One']
['And', 'I', 'said,', '"It']
['And', 'I', 'said,', '"It\'s']
```

Compare to the 5 hypotheses sampled with DPP ("to" was the top-1 in this case)

```
['And', 'I', 'said', 'to']
['And', 'I', 'said,', '"A']
['And', 'I', 'said,', '"One']
['And', 'I', 'said,', '"Wait']
['And', 'I', 'said,', '"You']
```

It is evident that DPP sampling found a much more diverse set of hypotheses for beam search. While four of the five beams in Top-k share the same semantic meaning for '"Un', DPP seems to diversify and only have two such beams with this meaning, instead adding in '"Wait' and '"You' using other contextual clues from the sentence. In this case, the model had already assigned high probabilities to these hypotheses as well, but they were simply crowded out of the top 5 hypotheses by those five similar words.

8.3 Runtime Analysis

In addition to the empirical runtime analysis completed in Fig. 1, where we observe that DPP-beam search runtime does not blow up with k, we perform theoretical runtime analysis on the top-k and k-dpp beam search algorithms. Here we assume running the decoder for a single step on a single word is a constant-time operation. Denote $|V|$ as the size of the vocabulary, m as the maximum sentence length, and k as the beam size.

Top k beam search: Computing scores for every word in the vocabulary, given a single hypothesis, and taking the top k scores, is a $O(|V|)$ operation. There are at most k hypotheses to continue from, and our beam search will continue for at most m steps, so the total runtime of beam search is $O(|V|k^m)$.

k-DPP beam search: Let S be the size of the initial set of top scores that we compute k-DPP from. Computing scores for every word in the vocabulary, given a single hypothesis, and taking the top S scores, is a $O(|V|)$ operation. Running the decoder once more to obtain feature vectors for each of these words is a $O(S)$ operation. Further, it is an $O(S^2)$ operation to compute $xx^\top$ where x is a $S \times 1$ vector. Our DPP-sampling is dominated by eigenvalue decomposition which is $O(S^3)$. Finally, our beam search will continue for at most m steps, so the total runtime of beam search is $O(|V|S^3k^m)$. When we set $S = \alpha k$ for some constant α, we get $O(|V|k^m)$.

In practice, our k-DPP beam search algorithm still does run slower than top-k beam search, but the worst-case runtime has the same bounds. This is promising considering that improvements in beam search like this one could scale to larger vocabularies, larger models, and larger beam sizes without any worry of exponential runtime blowup.

9 Conclusion

We found that DPPs had both the theoretical backing and real-world potential to improve diversity of beams within beam search. The worst-case runtime bounds are the same in both DPP and top-k beam search algorithms, provided that we first pool from the top ck hypotheses. This further supports DPP sampling as a viable improvement to beam search. Real-world analysis showed that pure DPP sampling underperformed top-k, but a hybrid approach seemed to track top-k performance. Qualitatively, increased beam sample diversity was observed when using DPP methods.

Ultimately, these results show that DPP is no drop-in replacement for top-k beam search, but when combined, they show potential to serve as a powerful situational tool for translating complex sentences.

Additional information

Our mentor is Ashwin Paranjape. Our project proposal and milestone was graded by Anand.

References

[1] Eungchun Cho. "Inner Product of Random Vectors on S^n". In: *Journal of Pure and Applied Mathematics: Advances and Applications* 9.1 (2013), pp. 63–68. URL: http://scientificadvances.co.in/admin/img_data/692/images/[4]%5C%20JPAMAA%5C%207100121137%5C%20Eungchun%5C%20Cho%5C%20[63-68].pdf.

[2] Chun-Wa Ko, Jon Lee, and Maurice Queyranne. "An exact algorithm for maximum entropy sampling". In: *Operations Research* 43.4 (1995), pp. 684–691.

[3] Alex Kulesza and Ben Taskar. "k-DPPs: Fixed-size Determinantal Point Processes". In: *Proceedings of the 28th International Conference on International Conference on Machine Learning*. ICML'11. Bellevue, Washington, USA: Omnipress, 2011, pp. 1193–1200. ISBN: 978-1-4503-0619-5. URL: http://dl.acm.org/citation.cfm?id=3104482.3104632.

[4] Alex Kulesza, Ben Taskar, et al. "Determinantal point processes for machine learning". In: *Foundations and Trends® in Machine Learning* 5.2–3 (2012), pp. 123–286.

[5] Chengtao Li. *dpp: Python implementation of DPP sampling*. URL: https://github.com/ChengtaoLi/dpp.

[6] Chengtao Li, Stefanie Jegelka, and Suvrit Sra. "Efficient sampling for k-determinantal point processes". In: *arXiv preprint arXiv:1509.01618* (2015).

[7] Kishore Papineni et al. "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.

[8] Yiping Song et al. "Towards a neural conversation model with diversity net using determinantal point processes". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[9] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems*. 2014, pp. 3104–3112.

[10] Ashwin K. Vijayakumar et al. "Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models". In: *CoRR* abs/1610.02424 (2016). arXiv: 1610.02424. URL: http://arxiv.org/abs/1610.02424.

[11] Yonghui Wu et al. "Google's neural machine translation system: Bridging the gap between human and machine translation". In: *arXiv preprint arXiv:1609.08144* (2016).