Sampford Beam Search

Noè Canevascini, Julius Gruber, Lothar Heimbach, Kevin Zhang
ETH Zürich
{noec, jgruber, hlothar, kezhang}@ethz.ch

Abstract

In machine translation we need a decoder which given probabilities over a dictionary generates a sentence with highest probability. Beam search is the most used algorithm for this task. It returns a set of k elements and their relative probabilities and gives us an intuition of the possible further evolutions of the sentence. However there are multiple issues associated with this algorithm. For example: as noted by (Cohen and Beck, 2019) large beam widths result in sequences of tokens with probability distributions associated with low evaluation scores. Another possible issue is the deterministic nature of this algorithm, which makes it unlikely to have diverse outputs. (Kool et al., 2019) developed a stochastic beam search (SBS) algorithm based on the Gumbel-Top-k trick and (Meister et al., 2021a) expanded on the idea replacing the sampling methods with Conditional Poisson Sampling (CPS). In this work we implement Sampford sampling and show that it can achieve a higher diversity at the same BLEU score when compared to beam search and other decoding methods.

1 Introduction

Machine translation is an area of natural language processing (NLP) which relies on conditional language modelling. Conditional language modelling, when used in machine translations, aims to find the probability of an output sentence in a target language y given an input sentence in a starting language x.

$$P(y|x)$$

A problem with this approach is that our output space $\mathcal{Y}$ grows exponentially with the sentence length n. As each position in the sentence could be occupied by some word in the target languages vocabulary V we get $|\mathcal{Y}| = |V|^n$. This makes machine translation a case of structured prediction as for large n, $|\mathcal{Y}| \to \infty$. This explosion in the cardinality of our output space makes it computationally infeasible to find the output sentence with the highest probability.

$$y^* = \arg\max_{y \in \mathcal{Y}} P(y|x)$$

To address this issue decoding algorithms are used to essentially restrict the number of sentences in the output space $\mathcal{Y}$ that get explored while obtaining a result close to y^*, providing a heuristic alternative to solve the conditional language modelling problem.

$$y^{**} = \arg\max_{y \in \hat{\mathcal{Y}}} P(y|x) \ : \ |\hat{\mathcal{Y}}| << |\mathcal{Y}|$$

The most widely used decoding algorithm is beam search. Beam search is a relaxed interpretation of greedy best-first search. Best-first search tries to find the path in a directed graph with the highest sum of the edges weights by selecting the node with the highest weight at every position. Applied to machine translation, this results in $|\hat{\mathcal{Y}}| = |V| * n$. Beam search expands on best-first search by selecting the k-best nodes at every position, where k is the beam-width, resulting in $|\hat{\mathcal{Y}}| = |V| * k * (n-1) + |V|$. Figure 3 is example of beam search with $k = 2$ and $V = \{A, B, C, D, E, END\}$ where END signifies the end of the sequence.

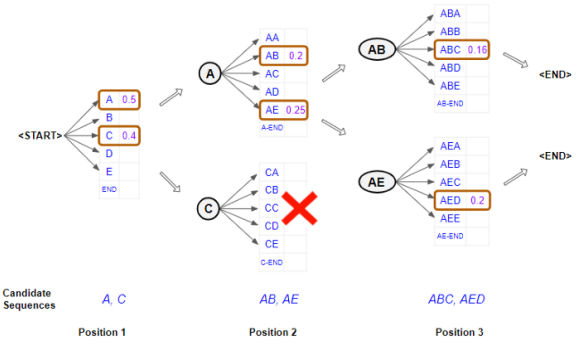

Figure 1: Visualization of beam search with beam-width 2 (Doshi, 2021)

A common drawback with beam search is that the output sentences tend to be very similar to each other resulting in a low diversity. Researchers like ((Meister et al., 2021a), (Kool et al., 2019), and (Shi et al., 2021)) developed stochastic versions of beam search in an attempt to increase the diversity of the output sentences. However, a higher diversity should not come at the cost of a worse translation. In this work we implement the sampling method Sampford (Tillé, 2006) and compare it's performance to other common decoding methods.

2 Related Works

In this section we discuss beam search and the different sampling methods currently used to try to make it stochastic.

2.1 Beam Search

The idea of using beam search in natural language processing was published in (Graves, 2012) or (Dept., 2015). The principle behind it is that instead of computing all possible conditional probabilities with all words (which would lead to prohibitively computational costs) we span a tree and only develop the most promising sub trees. Further details can be seen in 3.1. The ultimate objective is to find a sentence that maximizes the sequence model, which is a parametric distribution over sentences (Kool et al., 2019), (Kool et al., 2020).
This algorithm is deterministic and will always lead to the same output if given the same input. So the task is to make beam search stochastic. In 2.2 we give an intuition of the possible implementations.

2.2 Stochastic versions of Beam Search

(Kool et al., 2019) propose SBS. This implementation is based on what they call the Gumbel-Top-k trick. The algorithm relies on the Gumbel-Max trick as explored in (Gumbel, 1954) and (Maddison et al., 2014) with some slight modification such as sampling without replacement and taking a sample size of size k. The algorithm is explained more formally in 3.2
(Meister et al., 2021a) propose Conditional Poisson SBS (CPSBS). This method, as the name suggests, is based on the conditional Poisson sampling scheme (Hajek, 1964). CPSBS samples without replacement replacing the top-K operator with the conditional poisson sampling. The details of the sampling method are explained in 3.3.

3 Model

In this section, we give a formal introduction into beam search as well its stochastic pendants. As beam search has application beyond machine translation, we will consider it in a general framework. We base our setup on the previous work done in (Kool et al., 2019), (Meister et al., 2021a) and (Tillé, 2006).

3.1 Sequence Models

We work with a sequence model with bounded maximum sequence length $T > 0$. Formally, we consider a discrete probability space over the set of admissible outputs $\mathcal{Y} \subseteq V^T$ consisting of sequences over some vocabulary V. By filling up the words with a distinguished element of V, we assume that all outputs have the same length. We denote elements of $\mathcal{Y}$ as $y = (y_n)_{1 \leq n \leq T}$, and write $y_{<t}$ for the subsequence $(y_n)_{0 \leq n \leq t}$. The space $\mathcal{Y}$ is equipped with some probability distribution p. Usually, this distribution is determined by the distributions of the y_t conditioned on $y_{<t}$. The distribution of some random variable $Y \sim p$ thus admits the representation

$$p(Y_1 = y_1, \ldots, Y_t = y_t)$$

$$= p(Y_1 = y_1) \prod_{n=2}^{t} p(Y_n = y_n | Y_{<n} = y_{<n})$$

We then consider the maximization problem of p namely we are interested in some

$$y^* \in \arg\max_{y \in \mathcal{Y}} p(y)$$

As the size of $\mathcal{Y}$ is usually exponentially big, exact maximization is usually not computational feasible. Beam search is then used as a greedy-type approach that finds an approximate maximizer.
In this framework, beam search can be described as a sequence of sets $Y_1, \ldots, Y_T$ with Y_n consisting of sequences of length n. We denote the beam size by K. Additionally, for some $A \subseteq \mathcal{Y}$ and map $v : A \to R$ we introduce the notation $k - \arg\max_{a \in A} v(a)$ as the set of all $B \subseteq A$ of size k, such that there is no $C \subseteq A$ with a bijective map $f : B \to C$ such that for all $b \in B$ we have $v(b) \leq v(f(b))$, and there is $b \in B$ such that $v(b) < v(f(b))$. Intuitively, $k - \arg\max_{a \in A} v(a)$ contains all consists of all possible subsets of A with k elements that have the arguments for the k

highest values of v.
Then, beam search can be written inductively as

$$Y_1 = K - \arg\max_{v \in V} p(Y_1 = v)$$

and else

$$Y_n = K - \arg\max_{v \in V^n | v_{<n} \in V^{n-1}} p(Y_{\leq n} = v)$$

for $Y \sim p$. As long as $\mathcal{Y}$ is big enough, the output of beam search is $Y_T \subseteq \mathcal{Y}$.
The idea of stochastic beam search is to randomize the Y_n. Instead of finding the $k - \arg\max$ in each step, we sample such subset according to a predetermined probability distribution. By this, the procedure becomes a stochastic process. This way, the output can also be used to find an estimator for e.g. the expected BLEU score.

3.2 Sampling

In this subsection we discuss how the sampling in each time step may be implemented. There are different approaches that differ in the resulting distribution of the output as well as in the properties of the sample as estimator.
Since in our context the goal is to find a probable subset of K elements, we are only interested in sampling without replacement with fixed sample size. Generally, a sampling method over some population $U = \{y_1, ..., y_N\}$ with N elements is then simply given by a probability distribution on the space of all subsets with K elements. To simplify the notation, we represent also subsets by their indicator functions, namely for $Y \subseteq U$ we also write $Y_n = \chi_{y_n \in Y}$. Let

$$S_k = \left\{ s = (s_n)_{1 \leq n \leq N} \in \{0,1\}^N \mid \sum_{k=1}^{N} s_n = k \right\}$$

The sampling method is then a probability distribution Q on S_k.
There are different criteria on how to choose an appropriate distribution. For this, the inclusion probabilities π are important, which denote the probability of a given element being included in the sample set. Formally, we write $\pi(y_n) = Q(y_n \in Y)$, for a sample $Y \sim Q$. Note that π is no probability distribution over U, since

$$\sum_{n=1}^{N} \pi(y_n) = E^Q \left[\sum_{n=1}^{N} \chi_{y_n \in Y} \right] = k$$

where χ denotes the indicator function. If we use the sample in an estimator, we attempt to choose the distribution to optimize our estimate.
One approach would be to simulate a typical draw by draw procedure, picking one element in each step without replacing it. After an element is drawn, the remaining probabilities will need to be conditioned by renormalizing them. This approach was analyzed in the original setting of Stochastic Beam Search in (Kool et al., 2019).

3.3 Conditional Poisson Sampling

Another approach is the Conditional Poisson Sampling design, which is a special exponential sampling design on S_k. The distribution is then dependent by some parameters $w = (w_n)_{1 \leq n \leq N}$, given by

$$Q_{C,w,k}(Y) = \alpha_w^{-1} \prod_{n=1}^{N} w_n^{Y_n}$$

where $\alpha_{w,k} = \sum_{y \in S_n} \prod_{n=1}^{N} w_n^{y_n}$ is a normalizing factor that cannot be further simplified into an explicit form. We will later discuss an appropriate choice of w.
There is, however, a recursive method to sample efficiently, some typical methods can be found in (Tillé, 2006).
The inclusion probabilities can also not be found in close form. To find them, one can use fixed point iteration. Typically, there is a constellation of inclusion probabilities one is aiming for, the question is how to set the parameter w. However, this is very computational intensive, especially if this has to be done in every step of beam search. Instead, in (Meister et al., 2021b), they used an approximate implementation for Conditional Poisson Stochastic beam search. As it can be seen in (BONDESSON et al., 2006), if one to achieve some fixed given inclusion probabilities $(\pi(y_n))_{1 \leq n \leq N}$, the choice $w_n = \frac{\pi(y_n)}{1 - \pi(y_n)}$ gives a good approximation. In regard of the original maximization problem, the inclusion probabilities should then be proportional to $p(Y_{\leq n}) = v$ in the setup mentioned in 3.1.

3.4 Sampford Sampling

As the sampling design will have an impact on the algorithm viewed as generalized beam search as well as a statistical estimator, we analyze new design to evaluate whether they result in significantly different performances. The first additional design we consider is Sampford Sampling as found in (**?**). Here, the distribution is parametrized by inclusion probabilities $(\pi_n)_{1 \leq n \leq N}$ admitting $\sum_{n=1}^{N} \pi_n = k$.

By writing $\omega_n = \frac{\pi_n}{1-\pi_n}$, the

$$Q_{S,\pi,k}(Y) = \alpha_{\pi,k} \sum_{m=1}^{N} \pi_m Y_m \prod_{\substack{n=1 \\ n \neq m}}^{N} \omega_n^{Y_n}$$

where $\alpha_{\pi,k}$ is the corresponding normalizing factor. Although the representation of Sampford is more complex, and it is not an exponential method, Sampford admits an easy implementation as well. Furthermore, it bears the advantage that the inclusion probabilities can be exactly determined. Building on the code of (Meister et al., 2021a), we chose to implement Sampford sampling via the CPS rejective Sampford procedure, which is based on conditioning CPS of smaller sample size on a first sample according to a normalized version of the inclusion probabilities. The algorithm is given by

Algorithm 1: CPS rejective Sampford procedure

1: Sample some y according to the categorical distribution given by the $\left(\frac{\pi_n}{k}\right)_{1 \leq n \leq N}$
2: **repeat**
3: Sample $Y \sim Q_{C,\left(\frac{\pi_n}{1-\pi_n}\right)_{1 \leq n \leq N},k-1}$ as in Conditional Poisson Sampling
4: **until** $y \notin Y$
5: **return** $Y \cup \{y\}$

In line 3, we relied on the implementation of Conditional Poisson Beam Search in (Meister et al., 2021a), therefore. By this, we are implicitly using an approximate version as well.

4 Results

4.1 Experimental Setup

The code was implement within the fairseq library (Ott4 et al.). We used the `conv.wmt14.en-fr` model from (Gehring, 2017) and the `newstest2014` dataset which consists of 3003 sentences. [1] We built our code on the existing code bases from (Meister et al., 2021a), which itself builts upon (Kool et al., 2019). We decided to implement our code base in a mixture of Cython & Python. [2] The results show five different decoding algorithms, including Sampford, run on eight different entropy settings, by adjusting the sampling temperature from 0.1 to 0.8. For diverse beam search, the diverse beam search strength was adjusted from 0.1 to 0.8 to adjust the entropy setting. Our implementation of Sampford sampling only terminated in reasonable runtime for the minimum entropy setting with a sampling temperature of 0.1. We assume that this is due to the fact that in a higher entropy setting, our algorithm forms sentences of maximum sentence length regardless of the input sentence causing the infeasible runtime. Unfortunately we did not manage to find the bug responsible for this behavior in time and can only show the results for Sampford sampling with a sampling temperature of 0.1.

The diversity is defined as

$$d = \frac{1}{4} \sum_{n=1}^{4} d_n$$

where

$$d_n = \frac{\text{\# of unique n-grams in k translations}}{\text{\# of n-grams in k translations}}$$

and an n-gram refers to a sequence of n consecutive words in a sentence. For each sentence we translate from english to french we receive k translations. We calculate the diversity of these k translations and average it with the diversity of the other translations in the data set.

4.2 Results

We showcase our findings in two plots. We plot the BLEU score against the diversity for a beam-width of k = 10 and k = 20.

In figure 2 we can see that beam search has a lower diversity than our implementation of Sampford while achieving a similar BLEU score. Sampford also display a slightly higher BLEU score than diverse beam search for the same diversity. For the sampling temperature of 0.1 stochastic beam search and Sampford sampling are very close, with Sampford displaying a slightly higher BLEU score. It would be interesting to investigate in the future how the trend of Sampford and stochastic beam search compare as the sampling temperature increases.

Figure 3 shows an overall lower diversity when we increase the beam-width to 20 for all decoding methods. In comparison to figure 2 beam search loses its effectiveness in comparison to Sampford, as for a similar BLEU score Sampford achieves

[1] https://github.com/pytorch/fairseq/blob/main/examples/translation/README.md

[2] https://gitlab.ethz.ch/hlothar/sampford-sampling

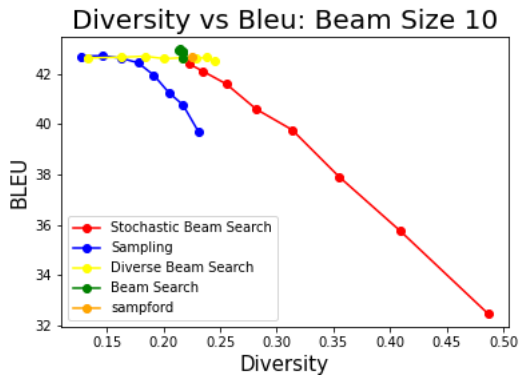

Figure 2: Average BLEU score versus diversity for sample size k = 10. Points correspond to different annealing temperatures 0.1, . . . , 0.8. Results for k = 5 show very similar trends.

a significantly higher diversity. The same can be said when comparing stochastic beam search to Sampford. Unlike for a beam-width of 10, Sampford achieves a significantly higher diversity score for a similar BLEU score indicating that Sampford is better suited for larger beam-widths than stochastic beam search and beam search. Diverse beam search and Sampford show a similar trend as they did for a beam-width of 10, with Sampford achieving a marginally higher BLEU score for the same diversity. In both figures sampling has the on-average lowest diversity score while not achieving a higher BLEU score than the other sampling methods, rendering this decoding method as the least preferment one for our application purposes.

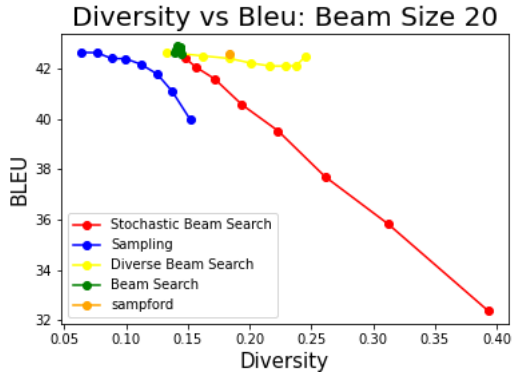

Figure 3: Average BLEU score versus diversity for sample size k = 20. Points correspond to different annealing temperatures 0.1, . . . , 0.8. Results for k = 5 show very similar trends.

5 Future Work

There are several ways this project can be extended. It would be interesting to implement the code in the current version of Fairseq so that it can be used by a wider audience of people. Furthermore, there would be the option to use more advanced algorithms. Another option would be to make the code run-able on a GPU or a multi-GPU setup. Although we implemented our code in PyTorch, we were not able to gain any performance boost, due to the sequential nature of the algorithm. The Monte-Carlo Tree Search algorithm, became famous in recent years in the deep learning community, with the use of it in the Alpha Go paper by (David Silver, 2017). This algorithm has not been applied to Beam Search in the NLP setting, but only has only been used for Games:(Cazenave).

6 Conclusion

In this paper we showed that Sampford sampling, a sampling-without replacement method for sequence models, achieves higher diversity scores for similar BLEU scores than other common decoding methods. We also showed that Sampford sampling is better suited to higher beam-widths, extending it's diversity advantage to other decoding methods.

References

LENNART BONDESSON, IMBI TRAAT, and ANDERS LUNDQVIST. 2006. Pareto sampling versus sampford and conditional poisson sampling. *Scandinavian Journal of Statistics*, 33(4):699–720.

Tristan Cazenave. *IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES*.

Eldan Cohen and Christopher Beck. 2019. Empirical analysis of beam search performance degradation in neural sequence models. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1290–1299. PMLR.

Karen Simonyan et alli. David Silver, Julian Schrittwieser. 2017. Mastering the game of go without human knowledge.

Carnegie-Mellon University.Computer Science Dept. 2015. Speech understanding systems: summary of results of the five-year research effort at Carnegie-Mellon University.

Ketan Doshi. 2021. Foundations of nlp explained visually: Beam search, how it works. [Online; accessed January, 2022].

Jonas Gehring. 2017. Convolutional sequence to sequence learning.

Alex Graves. 2012. Sequence transduction with recurrent neural networks.

E.J. Gumbel. 1954. *Statistical Theory of Extreme Values and Some Practical Applications: A Series of Lectures*. Applied mathematics series. U.S. Government Printing Office.

Jaroslav Hajek. 1964. Asymptotic Theory of Rejective Sampling with Varying Probabilities from a Finite Population. *The Annals of Mathematical Statistics*, 35(4):1491 – 1523.

Wouter Kool, Herke van Hoof, and Max Welling. 2019. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement.

Wouter Kool, Herke van Hoof, and Max Welling. 2020. Estimating gradients for discrete random variables by sampling without replacement. In *International Conference on Learning Representations*.

Chris J Maddison, Daniel Tarlow, and Tom Minka. 2014. A* sampling. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Clara Meister, Afra Amini, Tim Viera, and Ryan Cotterell. 2021a. Conditional poisson stochastic beam search.

Clara Meister, Tim Vieira, and Ryan Cotterell. 2021b. Best-first beam search.

Myle Ott4, Sergey Edunov, Alexei Baevskil, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. FAIRSEQ: A Fast, Extensible Toolkit for Sequence Modeling.

Kensen Shi, David Bieber, and Charles Sutton. 2021. Incremental sampling without replacement for sequence models.

Y. Tillé. 2006. *Sampling Algorithms*. Springer Series in Statistics. Springer.