System Description for the CommonGen task with the POINTER model

Anna Shvets
FabLab by Inetum
Paris, France
anna.shvets@inetum.com

Abstract

In a current experiment we were testing CommonGen dataset for structure-to-text task from GEM living benchmark with the constraint based POINTER model. POINTER represents a hybrid architecture, combining insertion-based and transformer paradigms, predicting the token and the insertion position at the same time. The text is therefore generated gradually in a parallel non-autoregressive manner, given the set of keywords. The pretrained model was fine-tuned on a training split of the CommonGen dataset and the generation result was compared to the validation and challenge splits.[1] The received metrics outputs, which measure lexical equivalence, semantic similarity and diversity, are discussed in details in a present system description.

1 Introduction

The 2021 edition of the Generation Evaluation and Metrics (GEM) challenge for the creation of living NLG benchmark leaderboard (Gehrmann et al., 2021), comprised four groups of tasks - summarization, structure-to-text, simplification and dialog. The CommonGen dataset makes part of the structure-to-text group and was designed to measure a common sense reasoning capacities of generative models given a set of concepts (Lin et al., 2020). Due to the nature of the constraint based text generation of the POINTER model (Zhang et al., 2020b) and resemblance in a generation strategy (the model takes a set of keywords as an input and generates a text, containing these keywords) the CommonGen dataset for hard constrained generation of the GEM benchmark appears to be a good fit for testing the model performance. The pretrained POINTER model was therefore fine-tuned on a training set of the CommonGen dataset and the inference results were compared to the validation and challenge splits of the same dataset.[2]

2 Data description and pre-processing

The Insertion-based transformer architecture leverage implies the use of the masking mechanism, the goal of which is to predict not only the likelihood of a token itself, but the likelihood of the token insertion between two given tokens, in other words, we need to predict the word and the place where a new word is inserted. In that regard, a text is preprocessed in a specific way, where the tokens are scored using a combination of three schemes of the token importance measurement (term frequency-inverse document frequency (TF-IDF), part-of-speech (POS) tagging and Yet-Another-Keyword-Extractor (YAKE)) and the highest scored tokens are replaced with a special no-insertion token [NOI] tag. This procedure is iterative and results in generation of several utterances out of the initial sentence. During the training phase, the model is initialised with the Multilingual BERT and its vobabulary is extended with the [NOI] tag. At the inference time, the masking mechanism is used in a reverse order, allowing an iterative tokens prediction - the model will chose to either generate a token or a [NOI] tag at a given generation stage and if the next stage contains [NOI] tag predictions only, the generation is finished.

The model was pre-trained on 12GB of Wikipedia corpora, therefore the pre-training data consisted of a well written English with the correct spelling, grammar and punctuation. For the fine-tuning, the sentences from the training split were preprocessed with the pre-training data generation script,[3] which inserts the token position masks in a gradual manner, resulting in a data augmentation from 67.389 source entries to 160.680 processed

[1]CommonGen have a private test set, which is not distributed by GEM benchmark, therefore a comparison to the test set was not possible.

[2]Available under the MIT license at https://github.com/dreasysnail/POINTER.

[3]Available in the project repository cited earlier.

entries.

3 Training details and decoding strategy

The fine-tuning was done on 8 cores (16GB of RAM each) of a TPU-v3 device, following the multiprocessing paradigm, and took three hours to train on 40 epochs with the batch size equal to 64 and gradient accumulation equal to 2. The finetuning hyperparameters were preserved from the original paper and included AdamW optimizer, learning rate equal to 1e-5, Adam epsilon equal to 1e-8, 10 warmup optimizer scheduler steps and the seed equal to 1.

The inference of the finetuned model was done using the concept sets from the validation and challenge splits of the CommonGen dataset. The decoding strategy included two sampling methods, applied separately - greedy and sampling. The greedy decoding is based on a greedy search algorithm, which consists of choosing the highest scoring token at a given time step, along with the temperature (Ackley et al., 1985), while sampling uses a combination of top-k (Fan et al., 2018), top-p (Holtzman et al., 2020) and the temperature parameters to render model predictions.

For the greedy decoding method, a temperature, which is a scale factor of each token's probability before going through softmax function, was set to its lower value 0.3, ensuring the most stable generations. This parameter alone draws a limit on the model's creativity, resulting in a more rigid generation.

For the sampling decoding method, the parameters promoting a high creativity of the model were chosen: the top-k window of the most probable tokens was set to 10, following the strategy expressed in the original paper (Fan et al., 2018), the top-p cumulative probability threshold for the most probable tokens was set to its highest tested value 0.95, according to the original paper (Holtzman et al., 2020), and the temperature was set to 0.9 - this is the highest lower probability threshould for this sampling parameter, allowing the maximum tokens pass-though without giving up stability of the text generation.

Other parameters were common for both sampling methods and included noi_decay and $reduce\_decay$, which were equal to 1, and $prevent$, $reduce\_stop$, $lessrepeat$, which were set to $true$. The inference for both decoding methods was done with the maximum sequence length equal to 256.

Description	Content
keys val.	ball court run throw
greedy val.	Olympic athlete then brings in the tennis ball straight back up down on the tennis court.
sampling val.	Olympic athlete quickly moves toward the soccer ball about halfway way up on the clay court.
target val.	The boy must run from one end of the court to the other to throw the ball into the hoop.

Table 1: Examples of generated text compared to the ground truth.

The opposite set of parameters (rigid versus creative) intended to explore the model's edge generative performance. This induces the metrics measurements for both types of the decoding strategy within validation and challenge splits.

4 Metrics outputs

Before diving in the metrics output results, let us explore a few examples of the generated text.[4] The Table 1 shows the examples of generation using greedy and sampling decoding methods for the validation split, compared to the human-generated target from the CommonGen dataset. To fairly measure the metrics output, the number of entries in the validation split was truncated to 500 in order to match the number of entries in the challenge set.

Since the goal of GEM challenge is an in-depth analysis of the model performance regarding lexical, semantic similarity and language richness, we will divide the analysis in separate subsections.

4.1 Lexical equivalence

The lexical equivalence was measured with four n-gram based automated metrics and is reflected in two tables: Table 2 and Table 3.

The Recall-Oriented Understudy for Gisting Evaluation (ROUGE), which relies on counting the matching n-grams in candidate and reference text, is a metric initially designed for evaluating summaries (Lin, 2004), which nowadays is widely used for many other tasks in natural language processing

[4]The complete lists of generated sentences along with the scripts for calculating the metrics can be found in a dedicated github repository: https://github.com/asnota/metrics

Sample	R1	R2	RL
greedy val.	0.137	0.008	0.109
sampling val.	**0.142**	0.008	0.106
greedy ch.	**0.142**	**0.009**	**0.111**
sampling ch.	0.136	0.008	0.103

Table 2: Lexical equivalence: ROUGE metric.

Sample	BLEU	NIST	METEOR
greedy val.	2.88	**0.114**	0.123
sampling val.	2.456	0.093	**0.141**
greedy ch.	**2.996**	0.113	0.125
sampling ch.	2.473	0.09	0.136

Table 3: Lexical equivalence: BLEU, NIST and METEOR metrics.

and generation. The ROUGE-1 (R1) and ROUGE-2 (R2) in a Table 2 reflects the co-occurrence of unigrams and bigrams in generated text versus validation or challenge splits of the CommonGen dataset. The ROUGE-L (RL) measures the longest in-sequence common n-grams and as we may observe, the values are quite small, meaning that the generated text might use different vocabulary, compared to the reference text. The ROUGE score is a bit higher for greedy decoding method of the challenge set.

While ROUGE is a recall-oriented metric, BLEU relies on a precision calculation of the overlapping n-grams and was primarily designed to measure the quality of the automatic translation (Papineni et al., 2002). The BLEU score augmentation is observed for the challenge set (Table 3), which might indicate, that the generated text might suffer from the noise, since it gives better scores when compared to a noisy reference text.

The calculation of the geometric mean with the BLUE score is completed by calculation of the arithmetic mean of the n-gram overlap with the NIST metric. This metric also calculates a degree of the informativeness of n-grams (rare n-grams are given more weight) and is less sensible towards small differences between the candidate and reference texts (Doddington, 2002). The NIST score shows no significant difference between validation and challenge splits, however the score itself is rather low, which indicates considerable lexical differences of the generated text compared to the reference text.

Additionally to the geometric and arithmetic mean, a harmonic mean of unigram precision and recall is calculated with the METEOR metric (Banerjee and Lavie, 2005). The advantage of this n-gram based metric is that the calculation includes synonym matching, stemming and word matching, which lowers the impact of alternative vocabulary and grammatical forms used in the generated text, compared to the golden human standard. Although the values appear to be low, it should be noted, that the maximum correlation with human judgement achieved was equal to 0.403.[5] The METEOR score is slightly higher for the challenge set and is generally higher for the sampling decoding method.

4.2 Semantic similarity

A recent shift towards neural based metrics changed the very essence of the metrics input - the words are represented by their embeddings, facilitating the calculation of many parameters, unavailable while calculating n-grams. In this system description three neural based automated metrics were used: BERTscore, which computes the cosine similarity of word embedding and applies greedy matching to maximize the similarity score in score arrays between words in the candidate and reference sentences (Zhang et al., 2020a), BLEURT, which uses a BERT model, pre-trained on a large amount of synthetic examples and finetuned on human judgement (Sellam et al., 2020), and NUBIA, which uses neural models output predictions on a set of parameters (Kane et al., 2020).

As shown in Table 4, there is no significant difference neither in BERTscore, nor in BLEURT score between validation and challenge sets. F1 and precision of the BERTscore are higher for greedy decoding, while recall is higher for the sampling decoding. We used the HuggingFace's API $load\_metric()$ from Datasets library to calculate the BLEURT score: by default, the API loads the BLEURT-base checkpoint with the sequence length limited to 128 tokens - the truncation of the original sentences resulted in an average score -1.4 for both decoding methods in both splits; the loading of the BLEURT-large checkpoint with the sequence length equal to 512, augmented the average score by 14%. The final values are shown in the above-mentioned Table 4 - the higher scores are observed for the greedy decoding method in both splits, however the overall values of the BLEURT

[5]Non-european languages have even lower METEOR scores - 0.347 on the Arabic data and 0.331 on the Chinese data, according to the ressource.

Samp.	F_{BERT}	P_{BERT}	R_{BERT}	BLEURT
g. v.	**0.842**	**0.822**	0.863	-1.233
s. v.	0.838	0.813	**0.865**	-1.252
g. ch.	**0.842**	**0.822**	0.863	**-1.225**
s. ch.	0.838	0.815	0.864	-1.243

Table 4: Semantic similarity: BERTscore and BLEURT.

Samp.	NUBIA score	semantic rel.
greedy val.	0.395	**0.803**
sampling val.	**0.523**	0.743
greedy ch.	0.406	0.35
sampling ch.	0.52	0.335

Table 5: Semantic similarity: NUBIA.

Sample	MSTTR	Dist1	Dist2
greedy val.	0.858	**0.19**	0.594
sampling val.	**0.88**	0.147	0.548
greedy ch.	0.858	**0.19**	**0.596**
sampling ch.	0.878	0.158	0.553

Table 6: Diversity: MSTTR and Distinct.

Sample	U1	U2	E1	E2
greedy val.	972	13115	5.818	10.241
sampling val.	**1285**	**20540**	**6.123**	**10.602**
greedy ch.	758	8172	5.788	9.638
sampling ch.	1030	11693	6.051	9.915

Table 7: Diversity: Unique and Entropy.

metric are rather low (since the maximum score that can be achieved with this metric is equal to 1), which indicates the semantic distance of the model's generations from the benchmark reference text.

NUBIA metric calculates such parameters as semantic relation, logical agreement, grammaticality, contradiction and a degree of new information presence (which might also signify the irrelevance) in the candidate sentence, regarding the reference sentence. In view of the current experiment's scope, we show the mean values of the cumulative NUBIA score and a semantic relevance measurement in Table 5. As we can see, the semantic relevance is considerably higher for the validation split.

4.3 Vocabulary diversity

Finally, the calculation of the lexical richness was done with four automated metrics - Mean Segmental Type-Token Ratio (MSTTR) (Johnson, 1944), Distinct (Li et al., 2016), Unique and Entropy (Shannon, 1948).

We can see in Table 6 that MSTTR is higher for the sampling decoding and is equivalent for greedy decoding in validation and challenge splits together. The Distinct score is surprisingly higher for the greedy decoding, but doesn't differ substantially between validation and challenge splits.

Table 7 shows that the amount of the unique unigrams and bigrams is higher for the sampling decoding (which is rather expected, as the sampling allows more creativity) and is substantially lower for the challenge set for both decoding methods. The Entropy is slightly higher for the sampling decoding method, and is generally higher for the validation set. This can be explained by the inconsistencies of the challenge set, which correlate with possible inconsistencies of the model generations, while a comparison with the perfect validation set, translates into higher rates of entropy, required to map one probability distribution to another.

5 Conclusions

The system description depicted the experiment on application of the CommonGen task from the GEM benchmark to a hard constraint text generation with the insertion based transformer. The use of eleven automated metrics for measuring the generative performance of the POINTER model allowed to detect the issues of the model output and reveal the advantages of a specific decoding method. For the lexical equivalence, METEOR metric seems to be the most relevant (since it takes stemmed forms of the words and makes the synonym comparison), when looking at the score augmentation for more creative text generations, accomplished with the sampling decoding method. The semantic similarity measured with the BERTscore and BLEURT neural based metrics showed that both validation and challenge splits result in a semantically equivalent text generations, with a small difference between decoding methods, while the application of NUBIA metric with a refined semantic relevance parameter resulted in a better score for the validation split. The Entropy showed the noisiness of the generated text for both decoding methods, and the Distinct score showed an unexpected boost for the greedy decoding, which means less words' repetitions than for the sampling decoding. Finally, the Unique score showed that sampling decoding method resulted in lexically richer text generations.

References

David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. 1985. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation.

Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Aremu Anuoluwapo, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna Clinciu, Dipanjan Das, Kaustubh D. Dhole, Wanyu Du, Esin Durmus, Ondřej Dušek, Chris Emezue, Varun Gangal, Cristina Garbacea, Tatsunori Hashimoto, Yufang Hou, Yacine Jernite, Harsh Jhamtani, Yangfeng Ji, Shailza Jolly, Mihir Kale, Dhruv Kumar, Faisal Ladhak, Aman Madaan, Mounica Maddela, Khyati Mahajan, Saad Mahamood, Bodhisattwa Prasad Majumder, Pedro Henrique Martins, Angelina McMillan-Major, Simon Mille, Emiel van Miltenburg, Moin Nadeem, Shashi Narayan, Vitaly Nikolaev, Rubungo Andre Niyongabo, Salomey Osei, Ankur Parikh, Laura Perez-Beltrachini, Niranjan Ramesh Rao, Vikas Raunak, Juan Diego Rodriguez, Sashank Santhanam, João Sedoc, Thibault Sellam, Samira Shaikh, Anastasia Shimorina, Marco Antonio Sobrevilla Cabezudo, Hendrik Strobelt, Nishant Subramani, Wei Xu, Diyi Yang, Akhila Yerukola, and Jiawei Zhou. 2021. The gem benchmark: Natural language generation, its evaluation and metrics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration.

Wendell Johnson. 1944. Studies in language behavior: A program of research. *Psychological Monographs*, 56(2):1–15.

Hassan Kane, Muhammed Yusuf Kocyigit, Ali Abdalla, Pelkins Ajanoh, and Mohamed Coulibali. 2020. NUBIA: NeUral based interchangeability assessor for text generation. In *Proceedings of the 1st Workshop on Evaluating NLG Evaluation*, pages 28–37, Online (Dublin, Ireland). Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. Commongen: A constrained text generation challenge for generative commonsense reasoning.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. 2020. Bleurt: Learning robust metrics for text generation.

Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. Bertscore: Evaluating text generation with bert.

Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. 2020b. Pointer: Constrained progressive text generation via insertion-based generative pre-training. In *EMNLP*.