Sixth Milestone Report: Nearest Neighbor Code Generation and Summarization

Alex Xie
https://axie66.github.io/07400-project/

Mentored by Vincent Hellendoorn
Institute for Software Research

April 4, 2022

1 Major Changes

There are no major changes since the last update.

2 Progress Report

2.1 Accomplishments

	Ruby	Javascript	Go	Python	Java	PHP
Base	16.01	16.24	19.74	19.43	20.29	26.23
Base + kNN	16.04	17.92	19.46	19.46	20.73	26.59
Base + Adv	15.54	16.16	19.41	19.21	19.98	25.89
Base + Adv + kNN	16.22	16.42	19.77	19.52	20.42	26.50

Table 1: BLEU results for summarization on CodeSearchNet, with and without adversarial training

While I found in my last report that kNN retrieval generally yielded improvements for code summarization, the uneven results suggested that not all languages were making equally good use of the datastore. Further, while I found that cross-lingual retrieval yielded improvements, I also hypothesized that this could be made more effective by explicitly pushing representations of similar examples in different languages closer together. Motivated by this, I have experimented with adding an adversarial objective in which the summarization model is trained jointly with a discriminator that predicts the input programming language given the model's kNN representations [1]. This encourages the model to learn language-invariant representations, which should in theory improve cross-lingual retrieval.

The results of these experiments are shown in Table 1 (rows 3 and 4 specifically show the new results for adversarial training). Adversarial training appears to hurt the base model, which is understandable as it somewhat constrains the model's expressiveness. However, the gains from kNN under adversarial training are fairly consistent and mostly make up for the base model's decrease in performance. Surprisingly, I find that adversarial training does not improve cross-lingual retrieval; while cross-lingual retrieval yielded +0.3 BLEU for the original model on Ruby, it yields only +0.1 with the addition of the adversarial objective. It may be that the optimal

balance between the two objectives is not currently being achieved; I plan on tuning it further to see if results improve.

Further, I hypothesize that retrieval may be especially effective in a zero-shot or unsupervised summarization setting; specifically, a setting where we have access to paired code-summary data in $n-1$ programming languages and unpaired code-only data in an nth language. This reflects a (somewhat) realistic scenario in which we have parallel data in a few high-resource PLs, but have difficulty acquiring such data for other languages. In my experiments, I take Ruby as the code-only language and use parallel data for the other 5 languages in the CodeSearchNet dataset.

I experiment with a modified training method for the base model in this altered setting: standard sequence-to-sequence training on the parallel data, masked language modeling (MLM) for the transformer encoder on the Ruby data, and adversarial training on the encoder's hidden states to ensure compatability between the encoder and the decoder (which is never trained on Ruby). In addition, as a baseline, I trained the base model from my original experiments, but without access to the Ruby data.

The results of these experiments are shown in Table 2. Interestingly, the base model trained without Ruby performs almost as well as the original model with access to Ruby. This may be due to the fact that the model has already been pretrained on Ruby data; further Ruby is relatively similar (both in syntax and distribution) to the other languages in the dataset. As such, it may be difficult to achieve meaningful results with the current CodeSearchNet setup. Hence, it may be necessary to create a small test set in a language that is very different from those that the model has already seen - for instance, Standard ML.

	Ruby	Javascript	Go	Python	Java	PHP
Seq2Seq w/o Ruby	15.96	16.23	19.68	19.58	20.35	26.25
Seq2Seq + Adv + Auto	15.60	16.22	19.43	19.29	20.17	26.32
Seq2Seq + Adv + MLM	-	-	-	-	-	-

Table 2: Language transfer experiments from the other five CodeSearchNet languages to Ruby. The MLM row is empty as experiments are still in progress. For the "Auto" row, I at first accidentally made the model trivially autoencode the Ruby input instead of doing masked language modeling; it somehow did pretty well.

2.2 Previous Milestone Goals

My main goals from the last milestone were investigating the adversarial training objective and exploring the unsupervised summarization setting, both of which I have made significant progress toward.

2.3 Surprises

The main surprise is the finding that the adversarial objective does not improve cross-lingual kNN retrieval; intuitively, since the representations are normalized across languages, it should be easier to retrieve across languages as well. I plan on investigating this further before the next milestone.

3 Next Steps

3.1 Looking Ahead

I mainly plan to further explore the unsupervised training setting described above. In particular, I have not yet fully fleshed out how kNN retrieval can be best utilized in this setting. Further, I plan on adapting this setting/method to autoregressive decoder-only code language models such as Polycoder [2], with the goal of adapting such a model to an arbitrary language with minimal additional training or performance loss.

3.2 Revisions to Future Milestones

Overall, the direction of my project has not changed since the last milestone; no future milestones need to be revised at this time.

References

[1] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations (ICLR)*, 2018.

[2] Frank F Xu, Uri Alon, Graham Neubig, and Vincent J Hellendoorn. A systematic evaluation of large language models of code. *arXiv preprint arXiv:2202.13169*, 2022.