Ariel Ekgren

Researcher building Large Language Models from Sweden.

Senior Research Scientist at the NLU group at AI Sweden.
Long standing interest in transformers and LLMs.

GPT-SW3: An Autoregressive Language Model for the Scandinavian Languages

Ariel Ekgren[1*], Amaru Cuba Gyllensten[1], Felix Stollenwerk[1], Joey Öhman[1], Tim Isbister[1], Evangelia Gogoulou[2], Fredrik Carlsson[2], Judit Casademont[1], Magnus Sahlgren[1]

[1]AI Sweden, Sweden

[2]RISE, Sweden

[*]Corresponding author: ariel.ekgren@ai.se

Our paper details the process of developing the first native large generative language model for the North Germanic languages, GPT-SW3.

We cover all parts of the development process, from data collection and processing, training configuration and instruction fine-tuning, to evaluation, applications, and considerations for release strategies.

We discuss pros and cons of developing large language models for smaller languages and in relatively peripheral regions of the globe, and we hope that this paper can serve as a guide and reference for other researchers that undertake the development of large generative models for smaller languages.

Introduction

There is a growing interest in building and applying Large Language Models (LLMs) for languages other than English.

This interest has been fuelled partly by the unprecedented popularity of ChatGPT that has propelled LLMs to the forefront of general awareness, and partly by the rapid commoditization of frameworks and infrastructure for training LLMs, which has drastically lowered the threshold for researchers to train and utilize LLMs.

However, even with the existence of accessible frameworks such as Hugging Face Transformers and commoditized compute infrastructure either through cloud or various national (and international) supercomputer initiatives, there are significant challenges to develop LLMs for smaller languages.

Introduction

The perhaps most obvious challenge is access to sufficient amounts of diverse, high-quality data. Apart from the basic question whether sufficient amounts of data at all exists for a smaller language, there may be additional complicating issues related to compliance with regulatory frameworks such as GDPR, the EU AI Act, and questions pertaining to copyright. We describe our data collection efforts in Section 3 (and in a separate paper, Öhman et al. (2023)).

Another challenge for prospective developers of LLMs is access to sufficient amounts of compute. Some countries have national compute infrastructure devoted to researchers, but such infrastructure may have limited GPU-resources, and access is typically regulated via specific allocation tiers, which may not be suitable for large-scale projects such as LLM training. On the other hand, cloud providers are typically always an option, but can be prohibitively costly.

Introduction

We have faced all of these challenges in our work on developing the first native LLM for the Scandinavian (or, more accurately, North Germanic) languages.

The LLM, which we call GPT-SW3, is a continuation of our previous Swedish-only model (Ekgren et al., 2022a).

GPT-SW3 is a collection of large decoder-only pretrained Transformer language models trained with a causal language modeling objective on a dataset containing approximately 320B tokens in Swedish, Norwegian, Danish, Icelandic, and English, as well as a set of 4 programming languages (Python, JavaScript, SQL and Shell script).

The suite of models ranges from 126M to 40B parameters, and instruction-tuned versions are also available for some of these models. This paper details the entire development process, from data collection and processing, training configuration and instruction-tuning, to evaluation and considerations for model release.

Data

The arguably most challenging aspect of building an LLM for a (set of) smaller languages is finding sufficient amounts of text data with sufficient quality and variety. Since there are no readily available large data collections for LLM pretraining in the North Germanic languages, we compiled our own training data, which we call The Nordic Pile (Öhman et al., 2023).

Our training data consists of text data collected from various open general data sources, such as MC4, OSCAR, OPUS, Wikipedia and parts of The Pile, as well as language-specific corpora such as the Norwegian Colossal Corpus, the Danish and Icelandic Gigaword corpora, and various data repositories, websites and discussion forums in Swedish. We also include a set of four different programming languages from the CodeParrot collection (Python, JavaScript, SQL and Shell script).

Data

We performed several steps of data processing on the collected data, including normalization, quality filtering and deduplication (both exact and fuzzy). The normalization takes care of non printing characters, and normalizes whitespace and Unicode characters. The quality filtering ap plies a set of heuristics inspired by Gopher and ROOTS (Rae et al., 2021; Laurençon et al., 2022), and the fuzzy deduplication utilizes MinHash LSH (Broder, 1997). Our training data, processing steps, and arguments for selection and filtering of sources is described in more detail in a separate publication (Öhman et al., 2023).

Data

We weight the different languages and categories (cf. Table 2) in such a way that the composition of the training data changes, while its total size stays the same. Note that this implies that some data are used multiple times while other data are discarded. More details can be found in App. A. After weighting, we end up with the following distribution of data in terms of languages:

- Swedish: 35.3%
- English: 23.4%
- Norwegian: 17.3%
- Danish: 14.8%
- Icelandic: 2.7%
- Code: 6.5%

Tokenizer

We employed the SentencePiece library to train a Byte-Pair Encoding tokenizer on a representative 1% sample of the model training data. The tokenizer has a vocabulary size of 64,000. Our reason for using a slightly larger vocabulary size compared to other LLMs (e.g. GPT-3, OPT, and GPT-NeoX have a vocabulary size of 50k tokens, while LLaMA only uses 32k tokens in the vocabulary) is that we want to improve the performance of the smaller languages included in our data, such as Icelandic.

Tokenizer

Our tokenizer works without explicit pre-tokenization. However, it splits digits and uses Sentence Piece's dummy prefix and the byte fallback feature. We also added repeated whitespace tokens (Black et al., 2022b) and special code tokens like <|python|> to the tokenizer's vocabulary, in order to improve the way code data is handled. Note that the special code tokens are present in the code data as well. We describe the tokenizer's features, training and evaluation in more detail in a separate paper (Stollenwerk, 2023).

After tokenization, our training data consists of around 320B tokens.

Training

We trained our models on 160 40GB A100 GPUs using the Nemo Megatron framework (Narayanan et al., 2021). We trained models of increasing size, starting out with the smaller models. This strategy was employed to identify problems with the pretraining procedure early on, before training the larger models.

All models have the same vocabulary size (64,000) and sequence length (2,048). The feed-forward dimension is always four times the embedding dimension. The models were trained using packing, meaning that each sample in a batch can consist of multiple documents delimited by end-of-text-tokens. We did not use attention-masking between documents.

Training

The learning rate schedule we employed is a function of the amount of data, as visualized in Figure 1. It is the same for all model sizes apart from a global factor, the maximum learning rate listed in Table 3. The training process starts with a short warm-up period that amounts to 0.5B tokens, during which the learning rate is increased from 0 to its maximum. Afterwards, we use a cosine decay to the minimum learning rate (1/10 of the maximum learning rate) for another 319.3B tokens. Finally, training is continued at a constant learning rate until up to 372.2B tokens are reached.

The validation loss during training can be seen in Figure 2. As expected, the larger models reach lower validation loss, and largely follow the expected scaling behavior (see Appendix B for more details). Contrary to some other works (Zhang et al., 2022), we did not experience any divergence during training, but we did observe the occasional gradient spike (with no catastrophic long-term effect).

Evaluation

Since we currently lack suitable evaluation benchmarks for generative language models in the North Germanic languages, we use language modeling perplexity on a set of held-out data to compare our models. We use character length normalization (Cotterell et al., 2018; Mielke, 2019) rather than token length for calculating perplexity formula, since token length favours tokenizers that use more tokens per sentence.

Evaluation

Task	0-shot						5-shot					
	126M	356M	1.3B	6.7B	20B	40B	126M	356M	1.3B	6.7B	20B	40B
ANLI Round 1	0.336	0.298	0.315	0.337	0.322	**0.368**	0.334	0.313	0.332	0.317	0.330	**0.350**
ANLI Round 2	0.317	0.338	0.345	0.333	0.343	**0.358**	0.341	**0.359**	0.340	0.332	0.333	0.350
ANLI Round 3	0.322	0.325	0.311	0.332	0.333	**0.391**	0.330	0.319	0.336	0.330	0.343	**0.364**
WSC	0.365	0.365	0.365	0.413	0.394	**0.548**	0.365	0.365	0.394	0.365	**0.519**	0.423
HellaSwag	0.279	0.322	0.393	0.457	0.502	**0.532**	0.280	0.321	0.387	0.452	0.504	**0.532**
Winogrande	0.493	0.517	0.571	0.617	0.632	**0.656**	0.522	0.527	0.557	0.607	0.657	**0.674**
PIQA	0.584	0.642	0.707	0.735	0.768	**0.772**	0.600	0.646	0.708	0.739	0.765	**0.776**
ARC (Easy)	0.388	0.408	0.549	0.609	**0.692**	0.687	0.412	0.473	0.588	0.647	0.707	**0.718**
ARC (Cha.)	0.204	0.200	0.253	0.294	0.352	**0.368**	0.191	0.213	0.276	0.312	0.360	**0.387**
OpenBookQA	0.140	0.186	0.214	0.220	0.268	**0.274**	0.148	0.188	0.228	0.260	0.240	**0.300**
HeadQA	0.224	0.236	0.266	0.278	**0.308**	0.302	0.227	0.243	0.273	0.295	0.233	**0.330**
Average	0.332	0.349	0.390	0.420	0.447	**0.478**	0.341	0.361	0.402	0.424	0.454	**0.473**

Table 5: LM Evaluation Harness accuracy scores of GPT-SW3 in 0-shot setting (left) and 5-shot setting (right). The best performance for each task and setting is marked in boldface.

Evaluation

Model	SE	DA	NO	EN
GPT-SW3 40B	**1.9240**	**1.8698**	**1.9270**	1.9660
GPT-SW3 20B	1.9458	1.8932	1.9491	1.9928
GPT-SW3 6.7B	1.9781	1.9229	1.9795	2.0152
GPT-SW3 1.3B	2.0665	2.0192	2.0741	2.1166
GPT-SW3 356M	2.1973	2.1568	2.2130	2.2477
GPT-SW3 126M	2.3748	2.3455	2.3992	2.4297
GPT-NeoX 20B	2.3807	2.3378	2.4245	1.9377
Falcon 40B	2.0194	2.2379	2.2705	**1.8152**
Falcon 7B	2.6546	2.7355	2.7740	1.8743
Falcon-RW 1B	3.7672	3.7187	3.7806	1.9765

Table 6: Evaluation of perplexity normalized on characters on held-out data for Swedish, Danish, Norwegian and English. The best score per language is marked in boldface.

Evaluation

Task	0-shot				5-shot			
	6.7B	6.7B-instruct	20B	20B-instruct	6.7B	6.7B-instruct	20B	20B-instruct
ANLI Round 1	**0.337**	0.329	0.322	**0.372**	**0.317**	0.309	0.330	**0.328**
ANLI Round 2	0.333	**0.376**	0.343	**0.381**	0.332	**0.341**	0.333	**0.359**
ANLI Round 3	0.332	**0.361**	0.333	**0.378**	0.330	**0.331**	0.343	**0.372**
WSC	**0.414**	0.385	**0.394**	0.365	0.365	**0.490**	**0.519**	0.375
HellaSwag	0.457	**0.503**	0.502	**0.528**	0.452	**0.502**	0.504	**0.527**
Winogrande	**0.617**	**0.617**	0.632	**0.639**	0.607	**0.616**	**0.657**	0.646
PIQA	0.735	**0.765**	**0.768**	0.764	0.739	**0.762**	0.766	**0.773**
ARC (Easy)	0.609	**0.679**	**0.692**	0.677	0.650	**0.686**	**0.707**	0.702
ARC (Cha.)	**0.294**	0.352	0.352	**0.355**	0.312	**0.360**	0.360	**0.382**
OpenBookQA	0.220	**0.274**	**0.268**	0.250	0.260	**0.288**	0.240	**0.282**
HeadQA	0.278	**0.318**	0.309	**0.310**	0.295	**0.334**	0.233	**0.323**
Average	0.421	**0.451**	0.447	**0.457**	0.424	**0.456**	0.454	**0.461**

Table 7: LM Evaluation Harness accuracy scores of our instruct models (6.7B and 20B) compared with their same-size base model counterparts in 0-shot setting (left) and 5-shot setting (right). The best performance for each task and setting is marked in boldface.

Evaluation

	GPT-NeoX	DaVinci	GPT-SW3 40B
ANLI Round 1	0.340	0.363	**0.368**
ANLI Round 2	0.343	**0.375**	0.358
ANLI Round 3	0.354	0.369	**0.391**
WSC	0.500	**0.548**	**0.548**
HellaSwag	0.535	**0.592**	0.532
Winogrande	0.661	**0.699**	0.656
SciQ	0.928	0.949	**0.955**
PIQA	0.779	**0.791**	0.772
ARC (Easy)	0.723	**0.762**	0.687
ARC (Challenge)	0.380	**0.435**	0.368
OpenBookQA	0.290	**0.336**	0.274
LogiQA	0.230	0.227	**0.290**
PROST	**0.296**	0.267	0.263
Average	0.489	**0.516**	0.497

Table 8: Comparison between GPT-NeoX 20B, OpenAIs DaVinci (175B), and GPT-SW3 40B model on LM Harness (0-shot). The best score for each task is marked in boldface.

Conclusion

This paper has detailed the development process for our family of North Germanic LLMs. It is at this point a perfectly reasonable question to ask why we at all should build a native LLM for a set of small languages with limited resources when the dominant LLMs of large corporations already can handle these languages in a reasonable (and often even superior) way. We have several answers to this question.

Conclusion

We believe that there is a desire and need for cultural and linguistic representativeness by informed choices and processing of data sources, transparency in all design choices and throughout the entire development process, democratizing access to natively built LLMs by open or hosted release, and open validation of model capacities as well as utilization of existing national compute infrastructure. Perhaps most importantly, the main goal of the GPT-SW3 initiative has been to give the Nordic research community full access to the weights of a native-language LLM, something that is not currently possible with other existing LLMs.

AI SWEDEN

Thank you for listening.