Efficient Methods for ML
Midterm Talk

Iason Kourouklis, Maarten Mäcking and Rickmer Weichenthal
Group: Not so Large Language Model

University of Hamburg

May 29, 2024

Language Characteristics

Brief description of *TinyStories*[1]

- short, english stories, using the vocabulary of a 3-4-year-old
- Idea: should contain the core elements of natural language (within a limited scope)
- generated by GPT-3.5 and GPT-4

For achieving a diverse dataset, each story should contain:

- a randomly picked noun, verb and adjective from a curated vocabulary
- a random subset of certain features (e.g., a bad ending)

[1] Ronen Eldan and Yuanzhi Li. "Tinystories: How small can language models be and still speak coherent english?" In: *arXiv preprint arXiv:2305.07759* (2023).

Cleanliness

Impurities in the dataset:

- encoding errors (e.g., "â€š" instead of ",")
- (few) typos (e.g., febuary, luggaage)
- stories without content [0.01%[2]]
- duplicates [15.13%[2]]
- non-ascii-symbols [0.16%[2]]

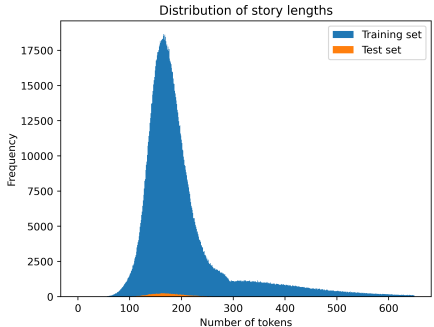
[2]Percentage of stories removed from the training set for this reason

Data

The data set consists of a training and test split
(we reserve 75K stories from the training set for validation)

Property	Training set	Test set
# of stories	1,796,422	21,956
# of tokens	370,067,106	4,419,913
# of unique tokens	42,801	11,244
avg. seq. length	206.0 (± 97.5)	201.3 (± 92.5)

- we use a max. seq. length of 256



Tokenization

Tokenizer: `basic_english` from `torchtext.data.utils`

- word-level tokenizer
- vocabulary: the 2,048 most common tokens in the training set

Tokenizer used in reference paper[3]:

- GPT-Neo tokenizer (10,000 most common tokens)

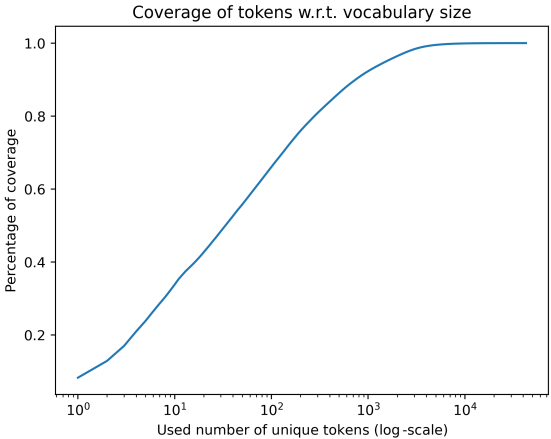
Why we opted for a word-level tokenizer:

- reduced sequence lengths
- higher interpretability
- model (hopefully) can only output valid words
- many tokens appear only a few times ($> 50\%$ of unique tokens appear ≤ 7 times)

[3] Ronen Eldan and Yuanzhi Li. "Tinystories: How small can language models be and still speak coherent english?" In: *arXiv preprint arXiv:2305.07759* (2023).

Tokenization

How many tokens can we represent with a given vocabulary size?



Embedding

using pytorch's `nn.Embedding` layer

- each token is represented by a dense vector
- words with similar meanings have similar representations
- $V \times d_{\text{model}}$ parameters for the embedding

Example: $V = 2048$, $d_{\text{model}} = 128$ (262,144 parameters)

Table: Most similar tokens to *mom* in embedding space ($n_{\text{layers}} = 4$, $d_{\text{ff}} = 512$)

Token	Cosine sim.
mommy	0.64615
mother	0.60687
mum	0.56941
mama	0.49344
grandma	0.47737

- $V \times d_{\text{model}} + V$ parameters for the unembedding

Positional Encoding

Sinusoidal positional encoding from the original transformer paper[4]

- very simple to compute
- no learnable parameters
- (slightly) improves the model's performance

$$PE(pos, 2i) = \sin\left(\frac{pos}{10,000^{2i/d_{\text{model}}}}\right)$$
$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10,000^{2i/d_{\text{model}}}}\right)$$

where pos is the position of the token and i the index of the embedding dimension.

[4] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

Positional Encoding

Figure: Training loss for model with and without positional encoding ($d_{\text{model}} = 192$, $n_{\text{layers}} = 6, n_{\text{heads}} = 6$ and $d_{\text{ff}} = 512$)

Evaluation: GPT-Eval

Use GPT-4 (UHHGPT) and Llama 3 (8B) for rating the stories in terms of

- grammar
- creativity
- consistency

Design criteria for optimizing the prompt:

1. rating should roughly match our rating
2. rating should be reliable (low std)
3. easy to parse format and fast computation time

Evaluation: ROUGE-N Score

ROUGE-N[5] score: Measure for the overlap of n-grams between generated stories and stories in the training set

Precision:

$$R_{n,p}(T_1, T_2) = \frac{\sum_{g_n \in T_1} Count_{match}(g_n)}{\sum_{g_n \in T_1} Count(g_n)}$$

F_1-measure:

$$R_n(T_1, T_2) = \frac{2R_{n,p}(T_1, T_2) \times R_{n,p}(T_2, T_1)}{R_{n,p}(T_1, T_2) + R_{n,p}(T_2, T_1)}$$

[5] Chin-Yew Lin. "ROUGE: A Package for Automatic Evaluation of Summaries". In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 74–81.

Evaluation: ROUGE-N Score

Procedure:

- Pick 100 stories $S_1, \dots S_{100}$ from the training set $\mathcal{S}$
- Cut stories in half and generate completions $T_1, \dots, T_{100}$
- Compare completion T_i with e.g. the original ending T_i'

Measure the following aspects[6]:

1. Novelty of the generated completion: $R_{2,p}(T_i, T_i')$
2. Similarity of the generated completions to each other: $\max_{j \neq i} R_2(T_i, T_j)$
3. Similarity to the most similar story in the training set: $\max_{S \in \mathcal{S}} R_{2,p}(T_i, S)$

[6] Ronen Eldan and Yuanzhi Li. "Tinystories: How small can language models be and still speak coherent english?" In: *arXiv preprint arXiv:2305.07759* (2023).

Transformer Model

Current architecture:

- $d_{\text{model}} = 192$
- $d_{\text{ff}} = 768$
- $n_{\text{heads}} = 6$
- $n_{\text{layers}} = 6$

Training time: 37.9min (RTX 4060 Ti)
Validation loss: 1.75

Further modifications:

- activation function: *GELU*[a] instead of *ReLU*
- (Start training with a high learning rate (e.g., 10^{-3}), lower during training)

[a] Dan Hendrycks and Kevin Gimpel. "Gaussian error linear units (gelus)". In: *arXiv preprint arXiv:1606.08415* (2016).

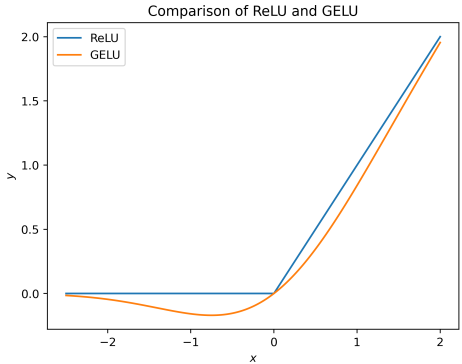


Figure: Plot of *ReLU* and *GELU*, where $GELU(x) = x \cdot \phi(x)$

RNN

Current architecture:

- $d_{\text{model}} = 256$
- $d_{\text{hid}} = 256$
- $n_{\text{layers}} = 2$

Training time: 8.1min (RTX 4060 Ti)
Validation loss: 2.54

Evaluation

Table: Avg. rating for the transformer

	GPT-4	Llama 3
Grammar	4.6 ($\pm$ 0.91)	4.9 ($\pm$ 0.94)
Creativity	5.4 ($\pm$ 1.00)	5.5 ($\pm$ 1.36)
Consistency	3.7 ($\pm$ 0.78)	7.1 ($\pm$ 0.94)

Table: Avg. rating for the RNN

	GPT-4	Llama 3
Grammar	2.9 ($\pm$ 0.53)	2.4 ($\pm$ 0.60)
Creativity	5.3 ($\pm$ 1.26)	6.0 ($\pm$ 1.48)
Consistency	2.1 ($\pm$ 0.44)	4.2 ($\pm$ 0.74)

Table: ROUGE-2 score for both models

	Transformer	RNN
$R_{2,p}(T_i, T_i')$	0.077 ($\pm$0.03)	0.065 ($\pm$0.02)
$\max_{j \neq i} R_2(T_i, T_j)$	0.096	0.118

Contextual Tracking

Table: Completed sentences by different transformer models. All models use 6 layers.

Beginning of story	2.9M	3.5M	12.2M
Alice was so tired when she got back home so she went	to take a nap.	to bed.	to the bedroom and lay down.
Lily likes cats and dogs. She asked her mom for a dog and her mom said no, so instead she asked	mom for a pet.	her dad to take her to the garage. Mom shakes her head.	her mom for a cat and some shoes.
Alice had both an apple and a carrot in her bag. She took the apple out of the bag and gave it to Jack. She reached into the bag again and took	the apple out of the bag.	a bite.	a big bite.

Project Vision

- Minimal goal: having a model that can generate stories which
  - are mostly grammatically correct
  - ideally make sense for the most part
  - differ by a significant amount from the training set
- Surprises: generated stories are already not bad :)
- Big vision: generate a (new) story which has a comparable quality to training stories
- Further procedure:
  - compare models of different sizes for evaluating which parameters are needed for achieving certain abilities of the model
  - compare our models to models in the TinyStories paper
- Obstacles: More parameters required for improving the model

Goals for this and next week

This week:

- Implementation of validation metrics
- Further preprocessing of the data
- Complete/improve RNN architecture

Next week:

- Try different ways of speeding up training (e.g., Automatic Mixed Precision)
- Optimizing the prompt for GPT-Eval (and automate Llama 3 pipeline)
- Try a different tokenizer (e.g., GPT-Neo)