NLP Semester Assignment "Over-Tokenized Transformer: Vocabulary is Generally Worth Scaling"

Συγγραφείς paper:

Hongzhi Huang, Defa Zhu, BangguWu, Yutao Zeng, Ya Wang, Qiyang Min, Xun Zhou

Students team:

Katsaidonis Nikolaos 03121868

Tzamouranis Georgios 03121141

Katsiadramis Kyriakos 03121132

Fotakis Andreas 03121100

Overview

Paper Analysis

Our Expirements

Project review

Overview

Paper Analysis

Our Expirements

Project review

Brief Summary of the Paper (1)

Main Idea: The paper investigates the effect of vocabulary size in a language model, both on **performance** and on **training cost**. It also examines **separating the input and output vocabularies** and how this affects those factors.

Over encoding (OE):
Use of n-gram tokens in the input (e.g., “cat”, “cat is”, “cat is sitting”). Instead of using a single token per word, we employ hierarchical n-gram tokenization and sum their representations.

Brief Summary of the Paper (1)

Main Idea: The paper investigates the effect of vocabulary size in a language model, both on **performance** and on **training cost**. It also examines **separating the input and output vocabularies** and how this affects those factors.

Brief Summary of the Paper (1)

Main Idea: The paper investigates the effect of vocabulary size in a language model, both on **performance** and on **training cost**. It also examines **separating the input and output vocabularies** and how this affects those factors.

1 **Over encoding (OE):**
Use of n-gram tokens in the input (e.g., “cat”, “cat is”, “cat is sitting”). Instead of using a single token per word, we employ hierarchical n-gram tokenization and sum their representations.

2 **Over decoding (OD):**
Larger output vocabulary. Multi-token output generation (e.g., “on the sofa” instead of three separate words).

Brief Summary of the Paper (1)

Main Idea: The paper investigates the effect of vocabulary size in a language model, both on **performance** and on **training cost**. It also examines **separating the input and output vocabularies** and how this affects those factors.

1 **Over encoding (OE):**
Use of n-gram tokens in the input (e.g., “cat”, “cat is”, “cat is sitting”). Instead of using a single token per word, we employ hierarchical n-gram tokenization and sum their representations.

2 **Over decoding (OD):**
Larger output vocabulary. Multi-token output generation (e.g., “on the sofa” instead of three separate words).

3 **Over-Tokenized Transformer(OTT):**
Combination of OE & OD to learn more complex dependencies.

Brief Summary of the Paper (2)

Brief Summary of the Paper (2)

Experiment 1 – Synthetic CFG Dataset (GPT-2):

- Training with OE (3-gram input tokens).
- Improved **accuracy & lower loss** with a larger input vocabulary.
- OD **harms smaller models**.

Brief Summary of the Paper (2)

Experiment 1 – Synthetic CFG Dataset (GPT-2):

- Training with OE (3-gram input tokens).
- Improved **accuracy & lower loss** with a larger input vocabulary.
- OD **harms smaller models**.

Experiment 2 – Downstream Tasks (PIQA, etc.):

- Tested on models with 151 M – 1 B parameters.
- OE-400 M → Same training cost as the 1 B baseline, but better performance.
- Up to 3.9× faster convergence, +1.3 % accuracy with OE + OD.

Brief Summary of the Paper (2)

Experiment 1 – Synthetic CFG Dataset (GPT-2):

- Training with OE (3-gram input tokens).
- Improved **accuracy & lower loss** with a larger input vocabulary.
- OD **harms smaller models**.

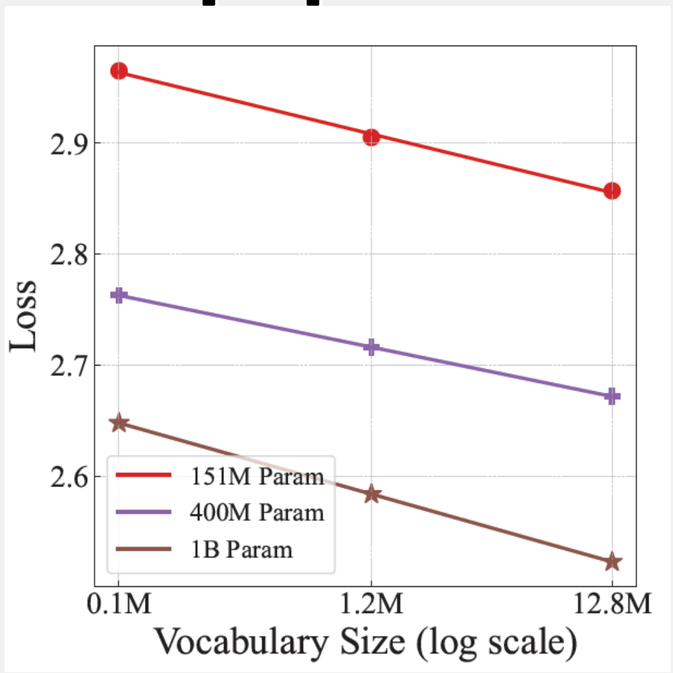
Experiment 2 – Downstream Tasks (PIQA, etc.):

- Tested on models with 151 M – 1 B parameters.
- OE-400 M → Same training cost as the 1 B baseline, but better performance.
- Up to 3.9× faster convergence, +1.3 % accuracy with OE + OD.

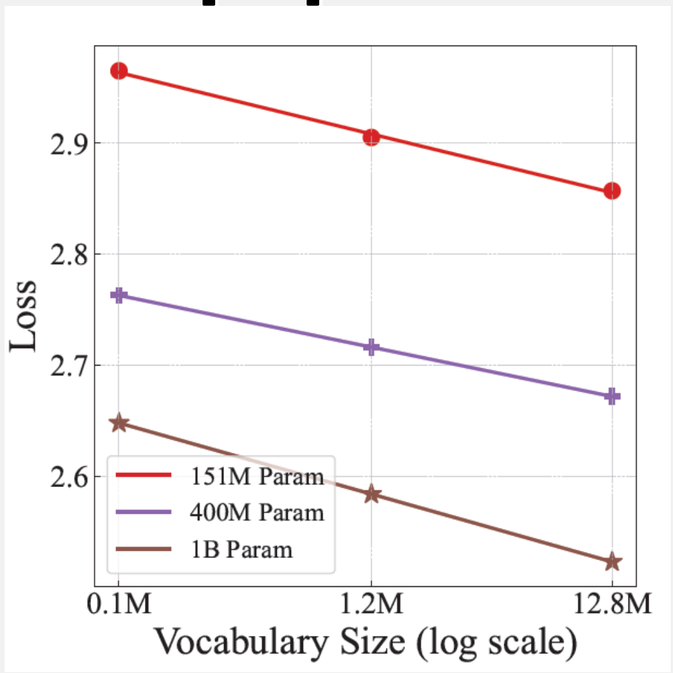
Findings:

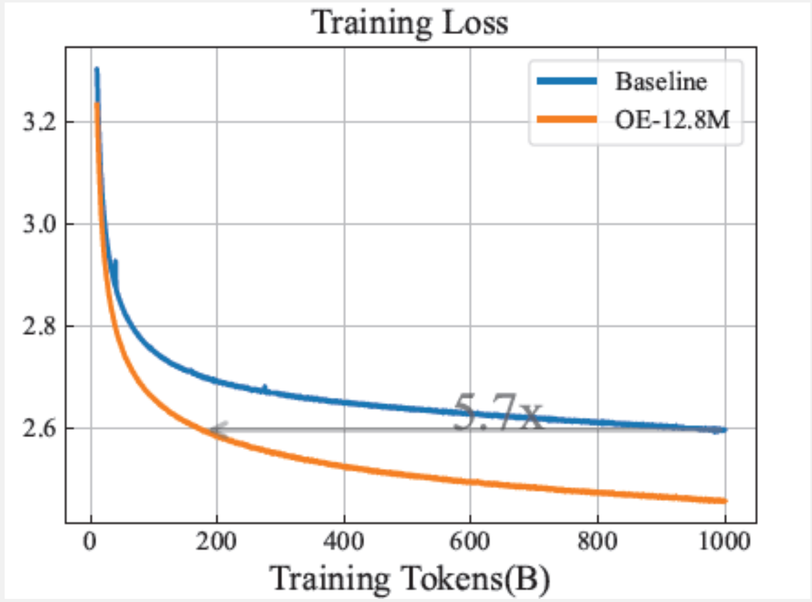
- A **larger input vocabulary** (via multi-gram embeddings) consistently **improves performance and reduces loss** regardless of model size (**log-linear relationship between vocabulary size and loss**).
- A **larger output vocabulary** can **harm smaller models.**

Some paper results

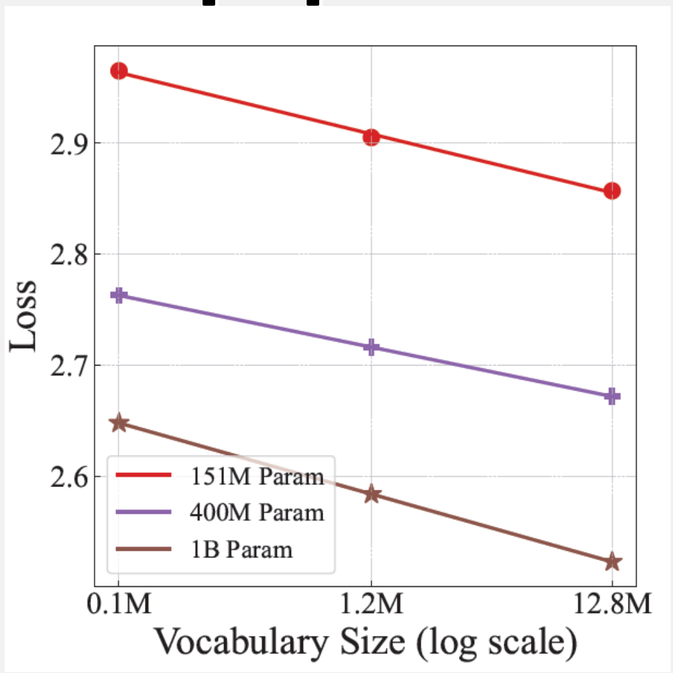


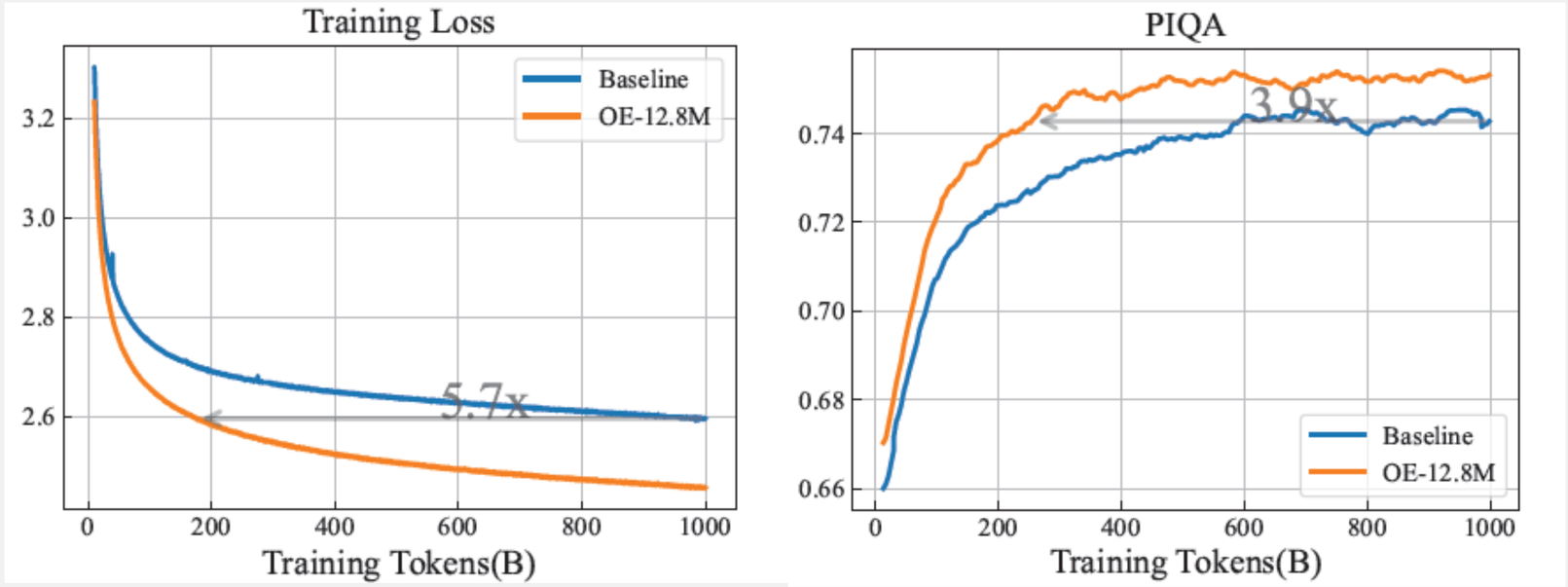
Some paper results





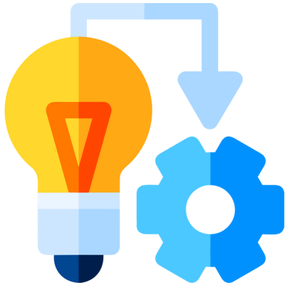
Some paper results





Overview

Paper Analysis



Our Expirements

Project review

Experiment 1: CFG Dataset (1)

Experiment 1: CFG Dataset (1)

Training with **GPT-2** (2.4 M parameters) as the **baseline** model without over-encoding, measuring **loss and generation accuracy** per epoch.

Experiment 1: CFG Dataset (1)

Training with **GPT-2** (2.4 M parameters) as the **baseline** model without over-encoding, measuring **loss and generation accuracy** per epoch.

Tested the **OE-m** model and compared its loss and generation accuracy to those of the corresponding baseline sizes.

Experiment 1: CFG Dataset (1)

Training with **GPT-2** (2.4 M parameters) as the **baseline** model without over-encoding, measuring **loss and generation accuracy** per epoch.

Tested the **OE-m** model and compared its loss and generation accuracy to those of the corresponding baseline sizes.

Increase m and compare the above metrics across various OE-m models.

Experiment 1: CFG Dataset (1)

Training with **GPT-2** (2.4 M parameters) as the **baseline** model without over-encoding, measuring **loss and generation accuracy** per epoch.

Tested the **OE-m** model and compared its loss and generation accuracy to those of the corresponding baseline sizes.

Increase m and compare the above metrics across various OE-m models.

Extract the corresponding **plots**.

Experiment 1: CFG Dataset (1)

Training with **GPT-2** (2.4 M parameters) as the **baseline** model without over-encoding, measuring **loss and generation accuracy** per epoch.

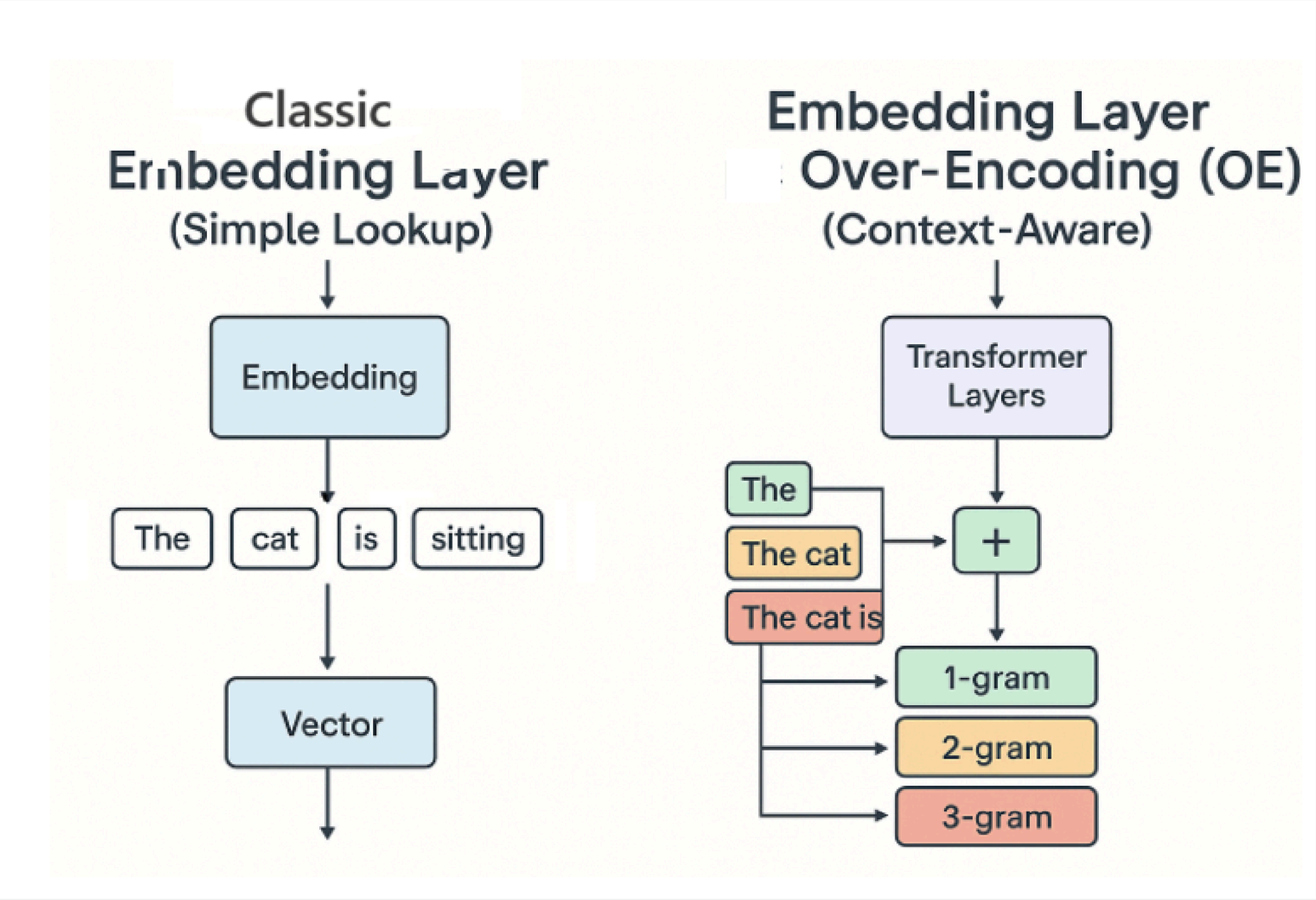
Tested the **OE-m** model and compared its loss and generation accuracy to those of the corresponding baseline sizes.

Note: The vocabulary of the cfg dataset is **{1,2,3,<PAD>}** so we have **4 1-grams, 4^2 2-grams και 4^3 3-grams.** As a result, we can not increase the vocabulary size logarithmically but just increase it and observ how this affects our metrics

Increase m and compare the above metrics across various OE-m models.

Extract the corresponding **plots**.

Visual representation of over encoding



Experiment 1: Datasets

Experiment 1: Datasets

Construction of a **synthetic dataset** with vocabulary = {1, 2, 3} using the following context-free grammar rules:

root |->20 21
root |->20 19 21
root |->21 19 19
root |->20 20

19|->18 16 18
19|->17 18
19|->18 18
20|->16 16
20|->16 17
20|->17 16 18
21|->18 17
21|->17 16
21|->16 17 18
21|->16 18

16|->15 15
16|->13 15 13
16|->14 13
16|->14 14
17|->15 14 13
17|->14 15
17|->15 14
18|->14 15 13
18|->15 13 13
18|->13 15

13|->11 12
13|->12 11 12
13|->10 12 11
14|->10 12
14|->12 10 12
14|->12 11
14|->10 12 12
15|->10 11 11
15|->11 11 10
15|->10 10
15|->12 12 11

10|->8 9 9
10|->9 7 9
10|->7 9 9
11|->8 8
11|->9 7
11|->9 7 7
12|->7 9 7
12|->9 8
12|->8 8 9

7|->2 2 1
7|->3 2 2
7|->3 1 2
7|->3 2
8|->3 1 1
8|->1 2
8|->3 3 1
9|->1 2 1
9|->3 3
9|->1 1

an example sentence

3322131233121131232113223123121111213211322311311
3223331231211121311331121321213333312322121312 32
2211112133221311311311311111132312331331333113 31
3333322312113111212211112112333123311211133133 33
3311233331311113333121132113121211333332121111 21
2132232233221332211322113232331311121322322322 1
2111333311213222213322112121331213313322122132 21
211213331232233312

Experiment 1: Datasets

Construction of a **synthetic dataset** with vocabulary = {1, 2, 3} using the following context-free grammar rules:

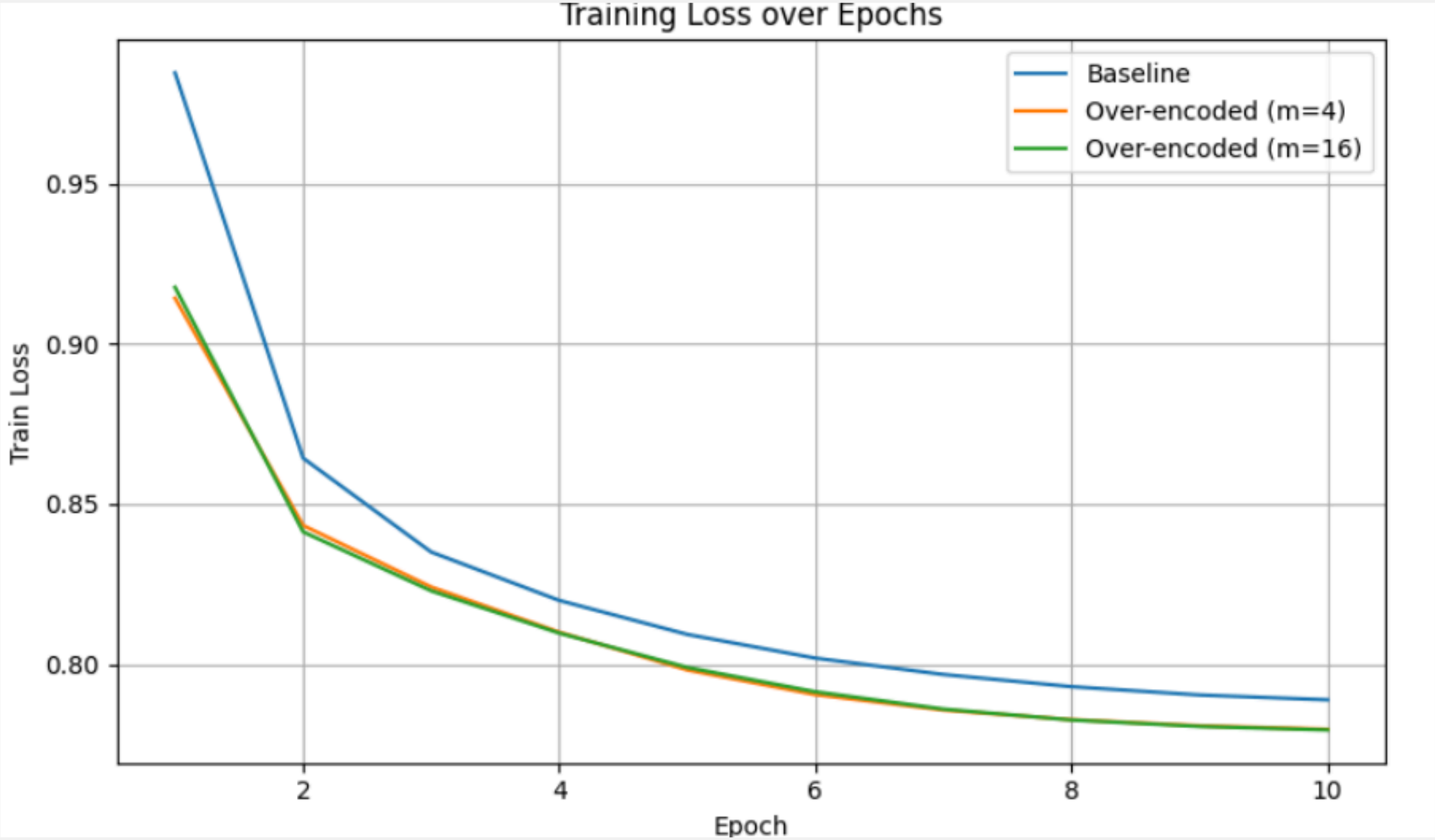
root |->20 21	19|->18 16 18	16|->15 15	13|->11 12	10|->8 9 9	7|->2 2 1
root |->20 19 21	19|->17 18	16|->13 15 13	13|->12 11 12	10|->9 7 9	7|->3 2 2
root |->21 19 19	19|->18 18	16|->14 13	13|->10 12 11	10|->7 9 9	7|->3 1 2
root |->20 20	20|->16 16	16|->14 14	14|->10 12	11|->8 8	7|->3 2
	20|->16 17	17|->15 14 13	14|->12 10 12	11|->9 7	8|->3 1 1
	20|->17 16 18	17|->14 15	14|->12 11	11|->9 7 7	8|->1 2
	21|->18 17	17|->15 14	14|->10 12 12	12|->7 9 7	8|->3 3 1
	21|->17 16	18|->14 15 13	15|->10 11 11	12|->9 8	9|->1 2 1
	21|->16 17 18	18|->15 13 13	15|->11 11 10	12|->8 8 9	9|->3 3
	21|->16 18	18|->13 15	15|->10 10		9|->1 1
			15|->12 12 11		

an example sentence

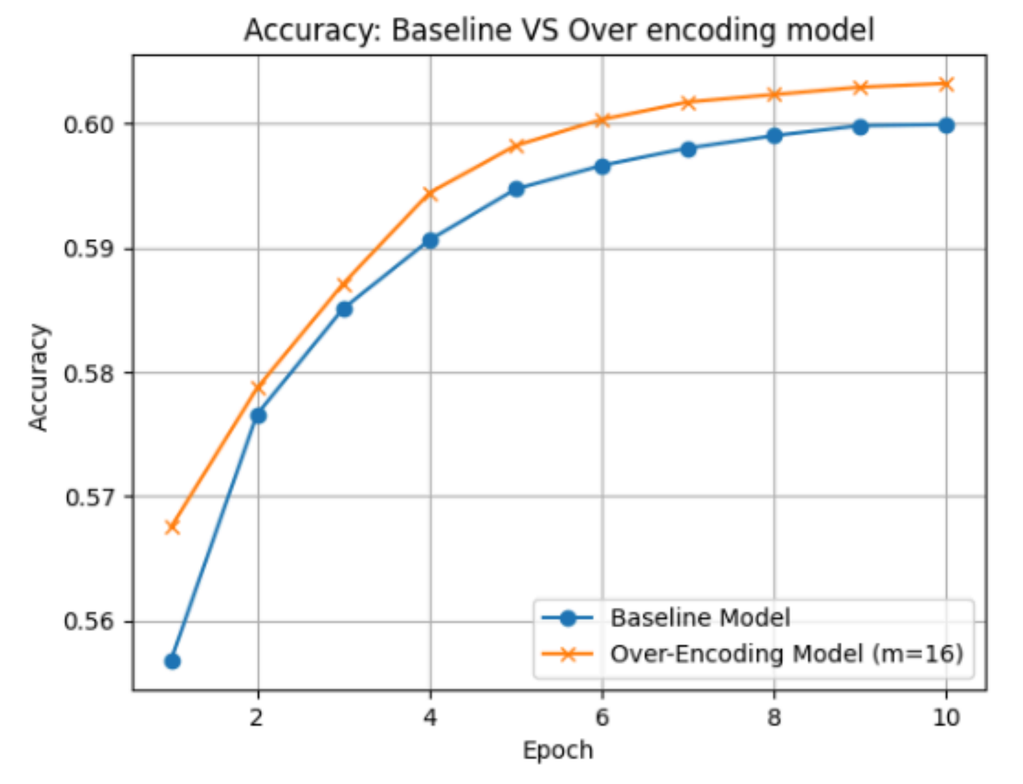
322131233121131232113223123121111213211322311311
322333123121112131133112132121333331232212131232
221111213322131131131131111113231233133133311331
333332231211311121221111211233312331121113313333
331123333131111333312113211312121133333212111121
213222322332213322111322113232331311121322322321
211133331121322221332211212133121331332212213221
211213331232233312

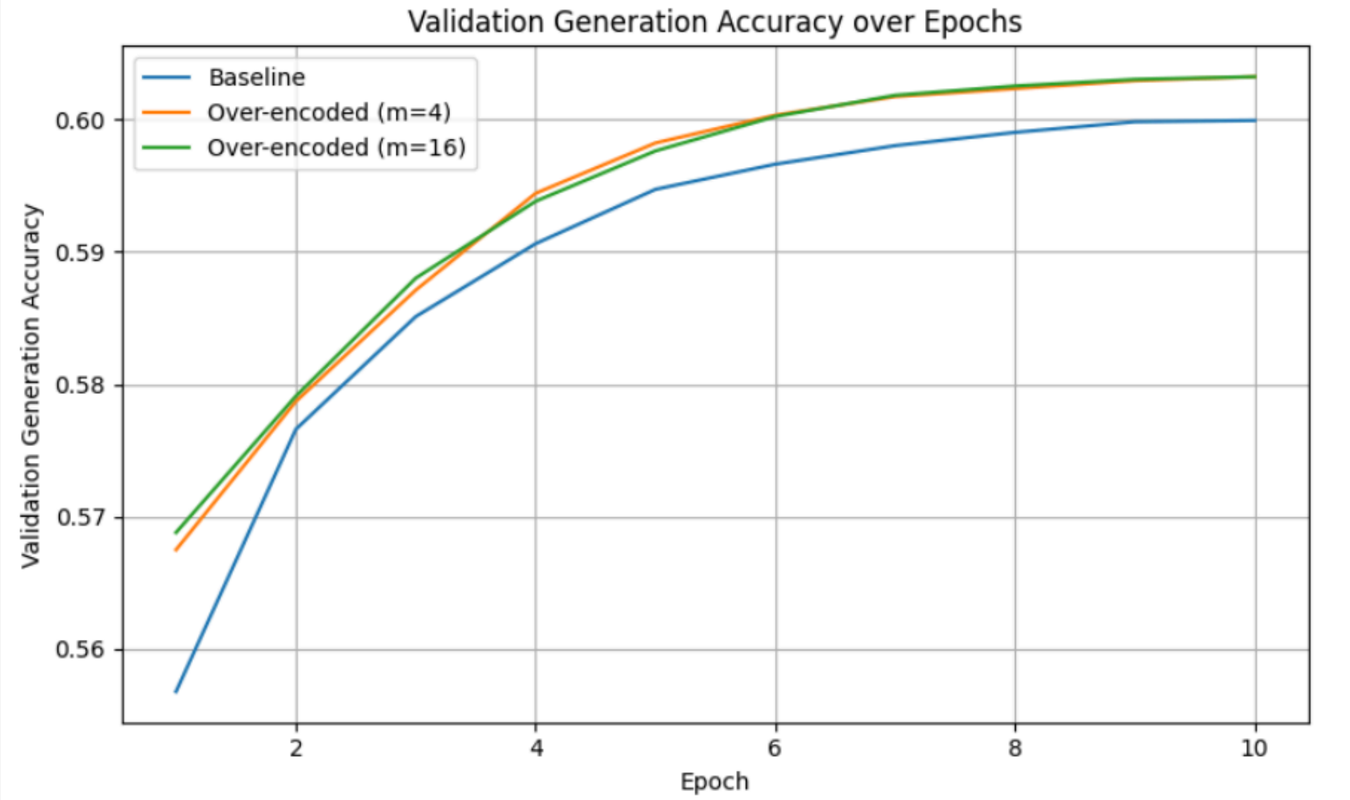
- **Total dataset:** 200,000 sequences of up to 729 characters.
- **Train/validation split:** 160,000 training sequences, 40,000 validation sequences.

Results (1)

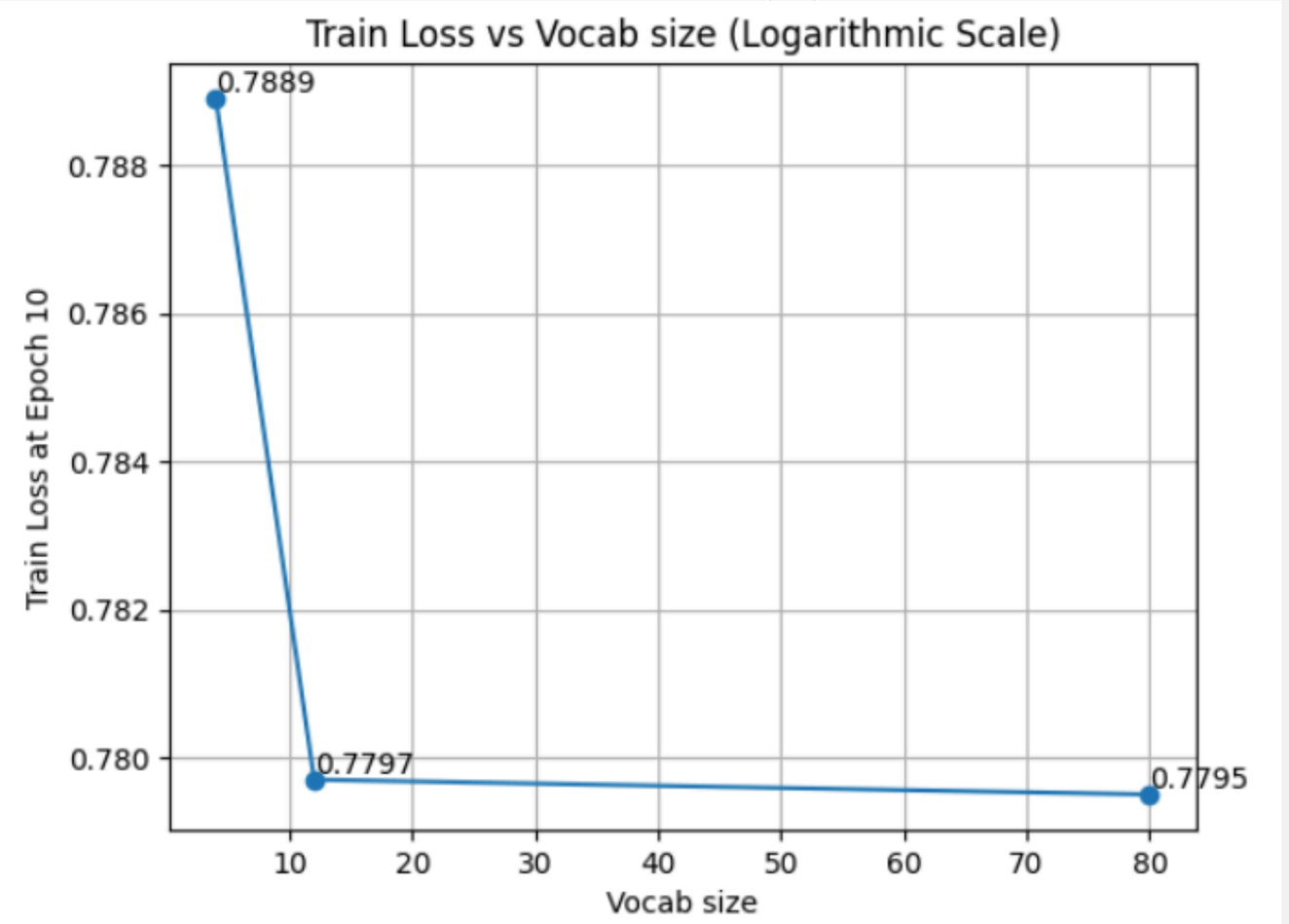

Training Loss over Epochs
Baseline
Over-encoded (m=4)
Over-encoded (m=16)
Train Loss
0.95
0.90
0.85
0.80
2
4
6
8
10
Epoch

Results (2)


Accuracy: Baseline VS Over encoding model
Accuracy
Epoch
0.56
0.57
0.58
0.59
0.60
2
4
6
8
10
Baseline Model
Over-Encoding Model (m=16)


Validation Generation Accuracy over Epochs
Validation Generation Accuracy
Epoch
0.56
0.57
0.58
0.59
0.60
2
4
6
8
10
Baseline
Over-encoded (m=4)
Over-encoded (m=16)

Results (3)



It is evident that with such a small vocabulary size, the reduction is very minor, since a logarithmic increase of m is not feasible for such a limited vocabulary.

Experiment 2 – Downstream Tasks

Experiment 2 – Downstream Tasks

Extract the most frequent n-gram combinations from a general text corpus (**WikiText**).

Experiment 2 – Downstream Tasks

Extract the most frequent n-gram combinations from a general text corpus (**WikiText**).

Load a **pretrained GPT-2** and freeze its parameters to preserve its general knowledge.

Experiment 2 – Downstream Tasks

Extract the most frequent n-gram combinations from a general text corpus (**WikiText**).

Load a **pretrained GPT-2** and freeze its parameters to preserve its general knowledge.

- Extend the embedding table with the n-gram combinations, initializing each entry as the average of its constituent embeddings.

Experiment 2 – Downstream Tasks

Extract the most frequent n-gram combinations from a general text corpus (**WikiText**).

Load a **pretrained GPT-2** and freeze its parameters to preserve its general knowledge.

- Extend the embedding table with the n-gram combinations, initializing each entry as the average of its constituent embeddings.

Training on the n-gram tokens while **keeping the 1-gram tokens unchanged** from the pretrained GPT-2.

Experiment 2 – Downstream Tasks (1)

Extract the most frequent n-gram combinations from a general text corpus (**WikiText**).

Load a **pretrained GPT-2** and freeze its parameters to preserve its general knowledge.

Logarithmically increase the n-gram vocabulary and compute the training loss.

- Extend the embedding table with the n-gram combinations, initializing each entry as the average of its constituent embeddings.

Training on the n-gram tokens while **keeping the 1-gram tokens unchanged** from the pretrained GPT-2.

Experiment 2 – Downstream Tasks (2)

Experiment 2 – Downstream Tasks (2)

HellaSwag dataset

- Testing the model on the HellaSwag task: given a **text as input** (context) and **four possible continuations**, one of which is logically coherent.

Experiment 2 – Downstream Tasks (2)

HellaSwag dataset

- Testing the model on the HellaSwag task: given a **text as input** (context) and **four possible continuations**, one of which is logically coherent.

PIQA dataset

- Testing the model on the PIQA task: **given a question as input** and **two possible answers**, one of which is correct.

Experiment 2 – Downstream Tasks (2)

HellaSwag dataset

- Testing the model on the HellaSwag task: given a **text as input** (context) and **four possible continuations**, one of which is logically coherent.

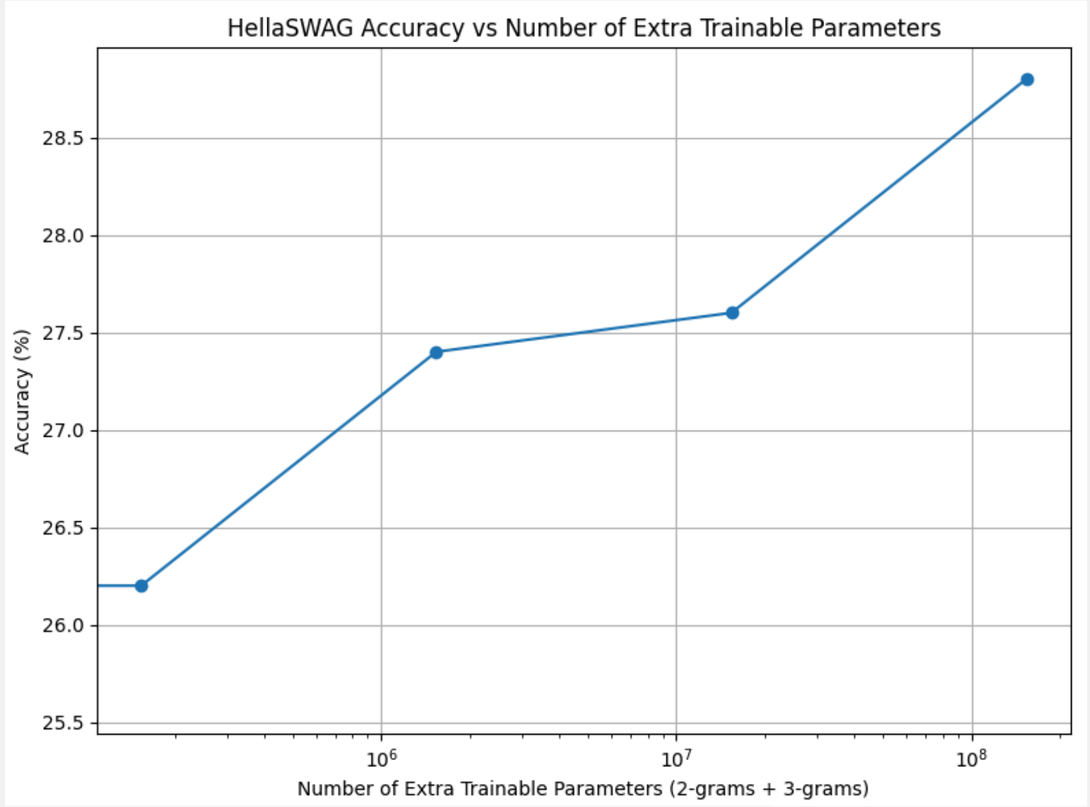
PIQA dataset

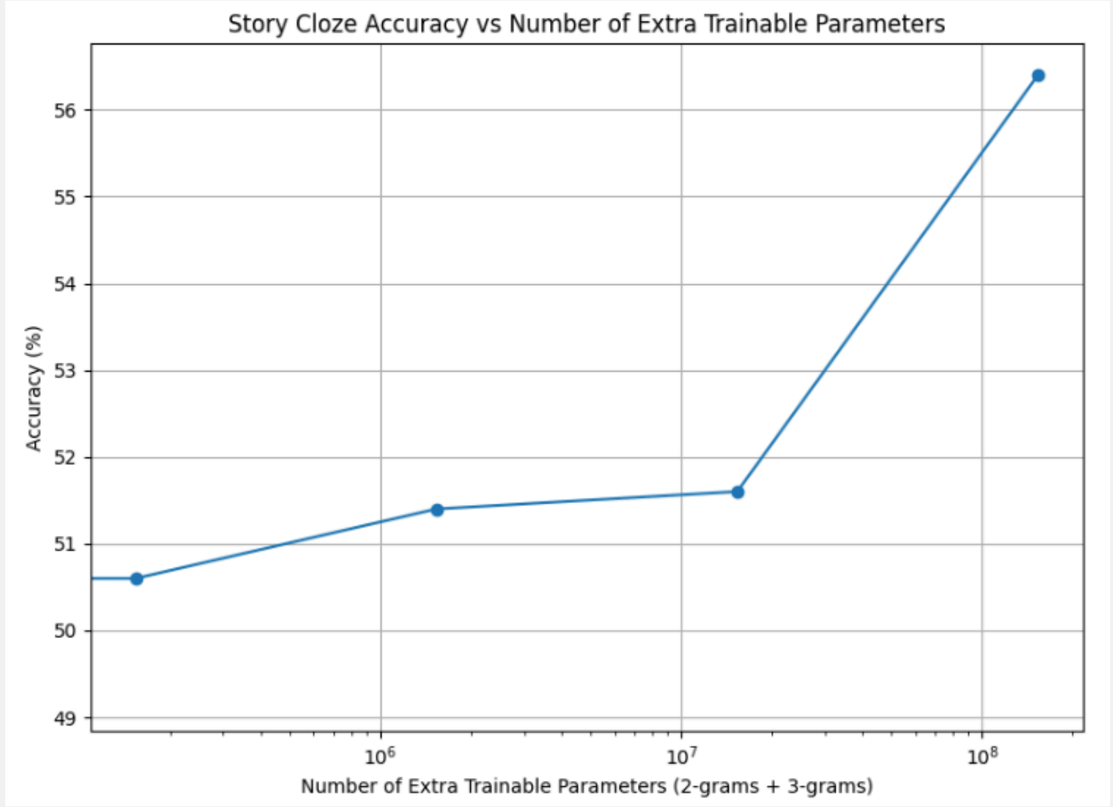
- Testing the model on the PIQA task: **given a question as input** and **two possible answers**, one of which is correct.

Story cloze dataset

- Testing the model on the Story Cloze task: input of **four context sentences** forming a story, and **two candidate endings**.

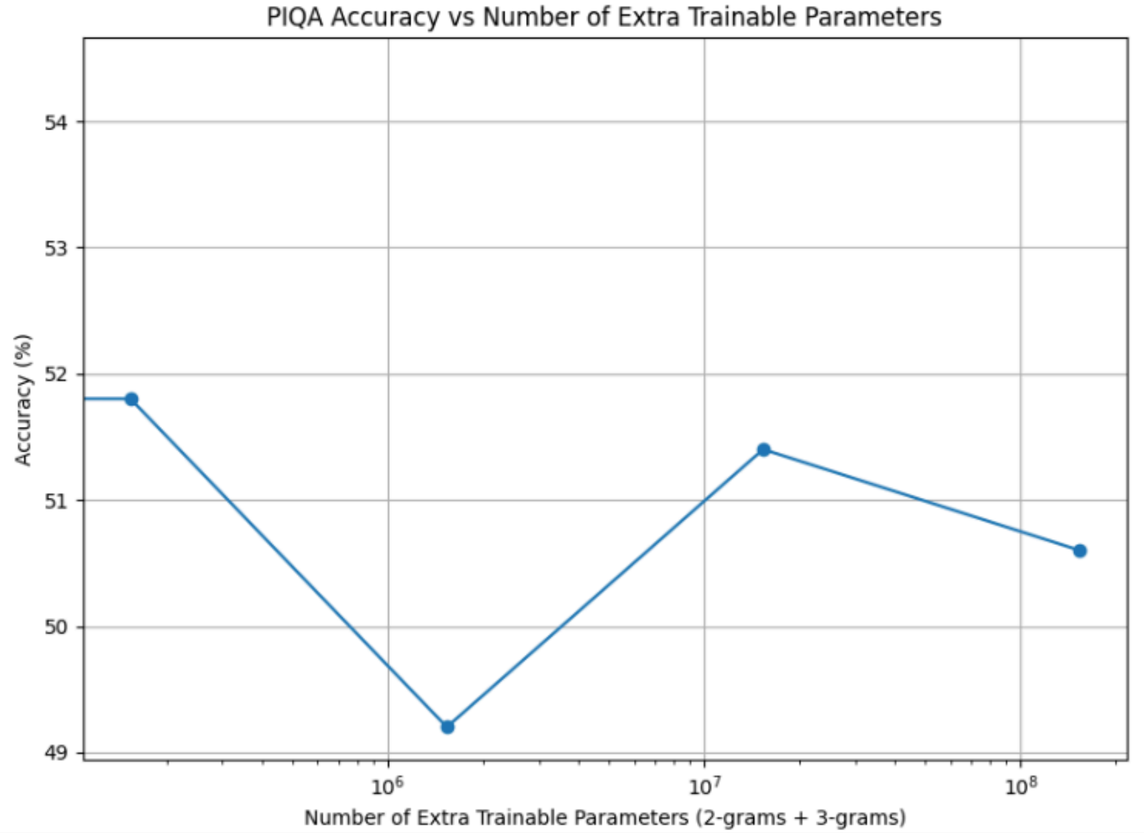
Results (1)

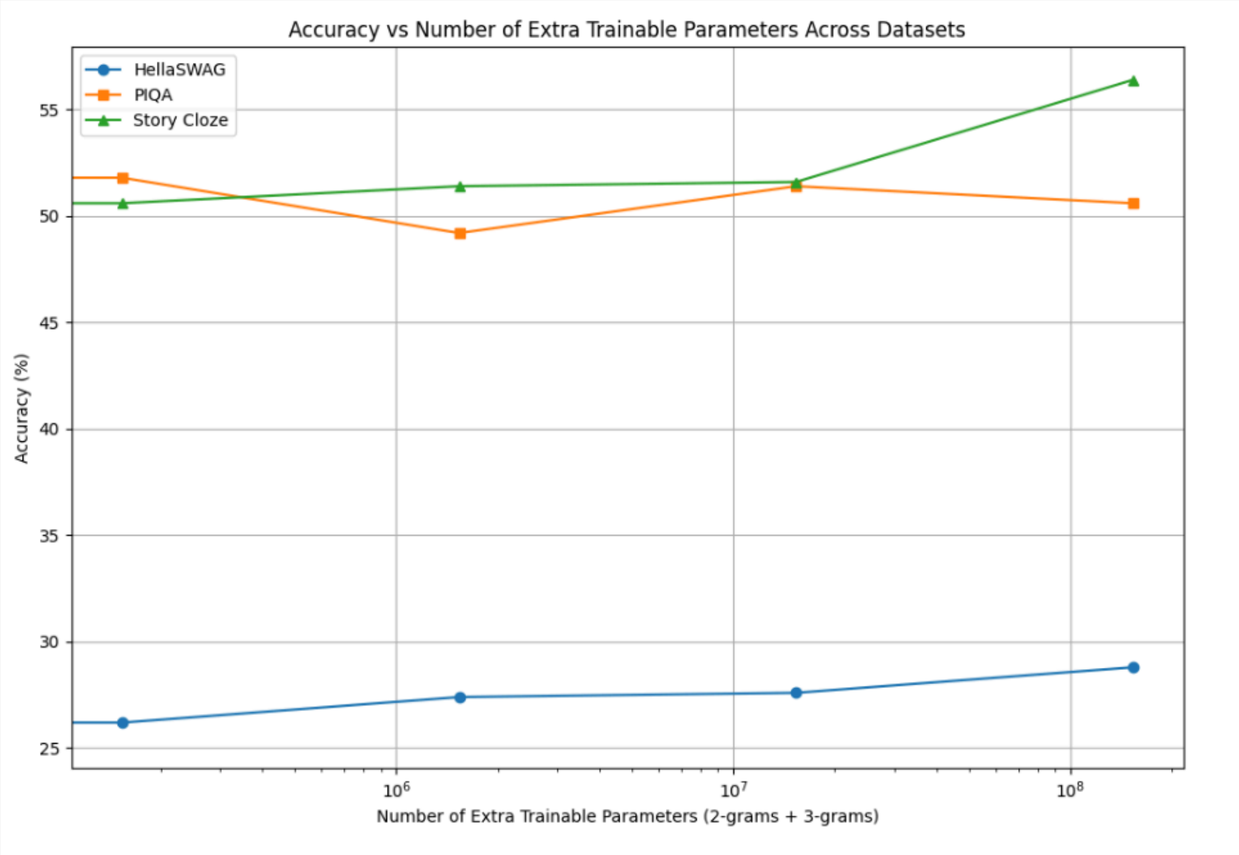




We observe that on context-creation tasks such **as HellaSwag or Story Cloze**, the model's **performance increases** as we increase the number of n-gram tokens.

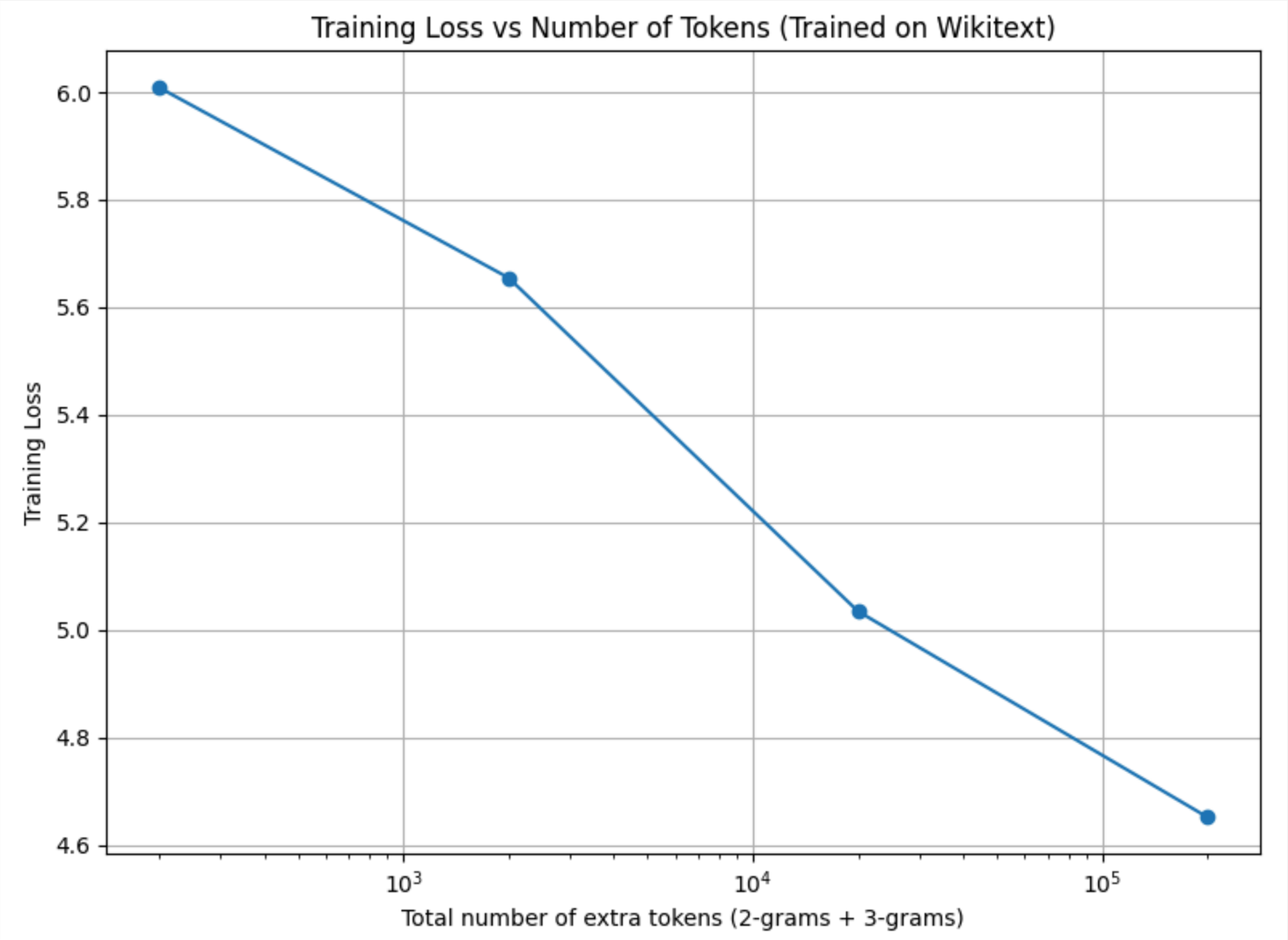
Results (2)





On the other hand, for datasets such as PIQA, adding n-gram tokens does not improve accuracy. Tasks like HellaSwag and Story Cloze align naturally with GPT-2's autoregressive training paradigm—so performance increases as we add more n-grams—whereas PIQA requires deeper reasoning and comparative judgment, capabilities that GPT-2 struggles with unless it is further fine-tuned on a question-answering dataset.

Results (2)



As shown in the paper, a **log-linear relationship between loss and the number of trainable tokens** is demonstrated.

Bonus Task: 4-gram Experiment

Bonus Task: 4-gram Experiment

- At this point, we will experiment by **increasing the vocabulary size even further**.

Bonus Task: 4-gram Experiment

- At this point, we will experiment by **increasing the vocabulary size even further**.

- Specifically, we will extend the embeddings with **4-grams and 5-grams** while simultaneously retaining the 2-grams and 3-grams.

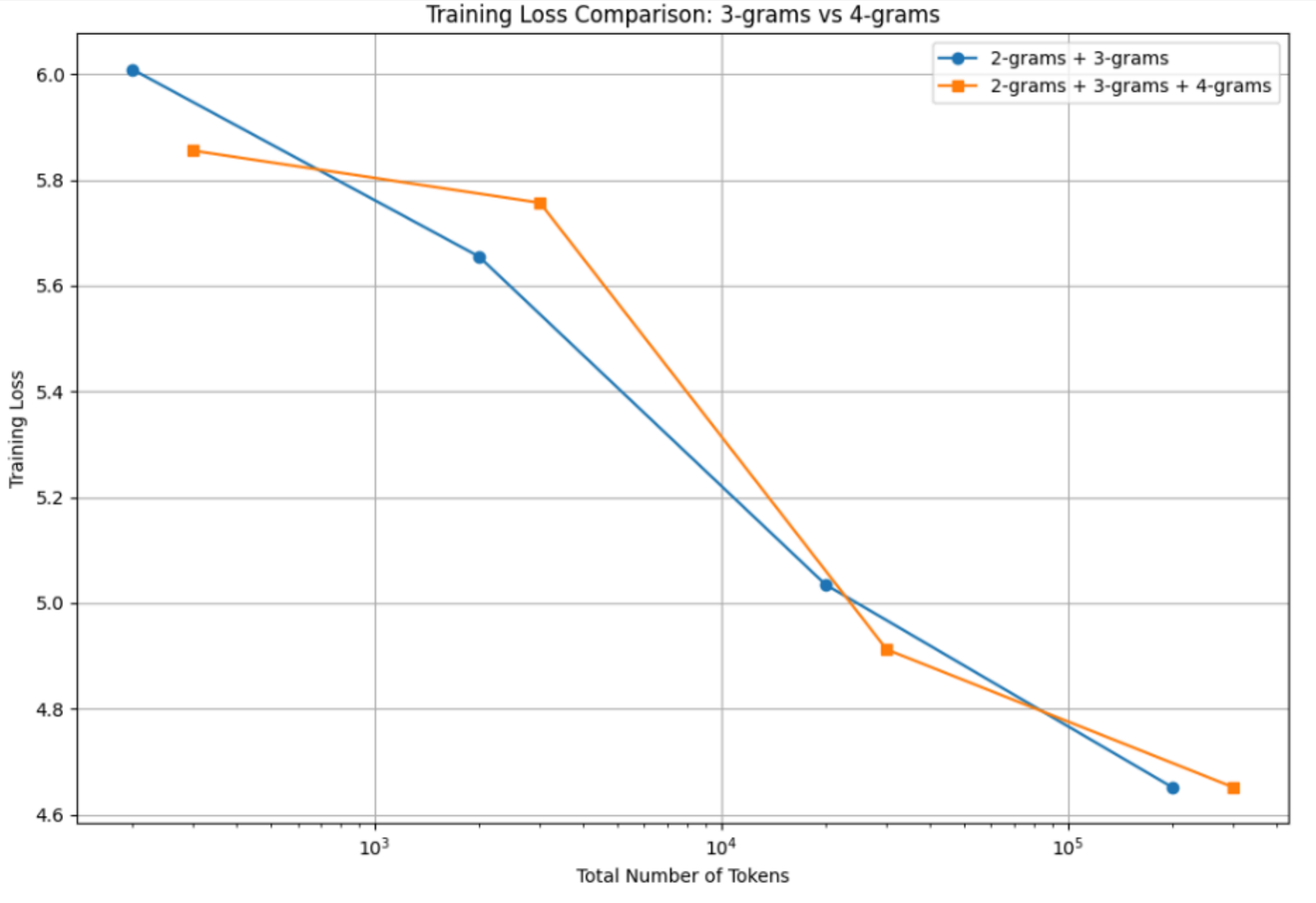
Bonus Task: 4-gram Experiment

- At this point, we will experiment by **increasing the vocabulary size even further**.

- Specifically, we will extend the embeddings with **4-grams and 5-grams** while simultaneously retaining the 2-grams and 3-grams.

- Our aim is to measure **how the training loss changed with the addition of extra embeddings,** as well as the system's **performance on downstream tasks** as described above.

Bonus Task: 4-gram expirement



- We do not observe a reduction in loss compared to the 3-gram models, nor any additional improvement in accuracy during testing.
- The same is observed for 5-gram models, and therefore **the computational complexity burdening training does not bring commensurate gains.**

Overview

Paper Analysis

Our Expirements

Project review

Evaluation and Future Extensions

Evaluation and Future Extensions

- Over-encoding steadily improves performance, continually increasing generation accuracy as the input vocabulary grows.

Evaluation and Future Extensions

- Over-encoding steadily improves performance, continually increasing generation accuracy as the input vocabulary grows.
- Reduction in training loss, enabling faster training (e.g., via early stopping).

Evaluation and Future Extensions

- Over-encoding steadily improves performance, continually increasing generation accuracy as the input vocabulary grows.
- Reduction in training loss, enabling faster training (e.g., via early stopping).

Future Extensions

- Dynamic vocabulary that evolves automatically during training, aiming to adapt to the data by dropping n-grams that do not “help” the model.

Evaluation and Future Extensions

- Over-encoding steadily improves performance, continually increasing generation accuracy as the input vocabulary grows.
- Reduction in training loss, enabling faster training (e.g., via early stopping).

Future Extensions

- Dynamic vocabulary that evolves automatically during training, aiming to adapt to the data by dropping n-grams that do not “help” the model.
- Experimentation with various n-gram combinations—not necessarily strictly hierarchical n-gram modeling—to determine the most effective configuration.

Task Distribution

Andreas Fotakis	Experimentation with downstream tasks, report writing, and presentation preparation.
Kyriakos Katsiadramis	Experiment with the CFG dataset, report writing, and presentation preparation.
Georgios Tzamouranis	Experimentation with downstream tasks, report writing, and presentation preparation.
Nikolaos Katsaidonis	Experiment with the CFG dataset, report writing, and presentation preparation.

Retrospective

Retrospective

Challenges and Future Improvements

Retrospective

Challenges and Future Improvements

- Difficulty loading the CFG dataset with 1 million samples. → On the fly data loading.

Retrospective

Challenges and Future Improvements

- Difficulty loading the CFG dataset with 1 million samples. → On the fly data loading.
- Slow model training. → Row-wise sharding for distributing the embedding table across GPUs, reducing communication cost.

Retrospective

Challenges and Future Improvements

- Difficulty loading the CFG dataset with 1 million samples. → On the fly data loading.
- Slow model training. → Row-wise sharding for distributing the embedding table across GPUs, reducing communication cost.

- GPT-2's weakness on question-answering tasks → Fine-tuning on a dedicated QA dataset instead of WikiText.

Retrospective

Challenges and Future Improvements

- Difficulty loading the CFG dataset with 1 million samples. → On the fly data loading.
- Slow model training. → Row-wise sharding for distributing the embedding table across GPUs, reducing communication cost.

- GPT-2's weakness on question-answering tasks → Fine-tuning on a dedicated QA dataset instead of WikiText.
- Experimentation with over-decoding techniques beyond over-encoding.

Thank you !

Students team:
Katsaidonis Nikolaos 03121868
Tzamouranis Georgios 03121141
Katsiadramis Kyriakos 03121132
Fotakis Andreas 03121100