Project 3: Using Pre-Trained Language Models

Language as Data at Göttingen University

Moujan Mirjalili, Farhan Kayhan, Merle (Joris) Hellwig 27.01.2025

Deliverable 1: Task Analysis for SST-2 (Stanford Sentiment Treebank)

Task Overview

- **Task Name:** SST-2 (Stanford Sentiment Treebank)
- **Dataset Source:** Sentences extracted from movie reviews on Rotten Tomatoes
- **Total Sentences:** 11,855 single sentences
- **Unique Phrases:** 215,154 phrases in the parse trees of 11,855 sentences
- **Annotation:** Each phrase annotated by 3 human judges
- **Task Type:** Binary sentiment classification.
- **Objective:** Predict whether the sentiment of a given sentence is positive or negative.

Task Setup with Examples

- **Input**: A single English sentence.
- **Output**: A binary label (0 for negative, 1 for positive).
- **Examples:**
  1. **Input**: A fascinating look at a deeply troubled man. → **Output**: Positive (1).
  2. **Input**: A dull and lifeless movie. → **Output**: Negative (0).

Relevant Statistics

Using publicly available data from the GLUE benchmark for SST-2:

- **Original Dataset Size:**
  1. **Training Set**: 67,349 examples.
  2. **Validation Set**: 872 examples.
  3. **Test Set**: 1,821 examples (labels withheld for competition submissions).
- **Our Dataset Size:**

  As the Test Set is Private we splitted the Train Dataset in Validation and Training Dataset and used the Old Validation Dataset as a test Set
  1. **Training Set:** 57178 examples.

2. **Validation Set:** 2384 examples.
3. **Test Set:** 872 examples.

- **Original Class Distribution:**
  1. **Positive Sentiment**: ~54% of the dataset.
  2. **Negative Sentiment**: ~46% of the dataset.

  The class distribution is relatively balanced, minimizing bias in evaluation.
- **Our Class Distribution:**

  We rebalanced the Training and Validation Dataset to 50%
- **Sentence Length:**
  1. **Average Sentence Length**: ~19 words.
  2. **Longest Sentence**: ~52 words.
  3. **Shortest Sentence**: 2 words.
- **Preprocessing:**

  The sentences are already tokenized and lowercased. However, we added a new split as described previously.

Manual Instance Analysis

We reviewed a subset of sentences to classify them into **easy** and **difficult** based on linguistic and contextual complexity.

- **Easy Instances (10 Examples):**
  1. **Input**: A thrilling and suspenseful movie! → **Output**: Positive (1).
  2. **Input**: Completely boring. → **Output**: Negative (0).
  3. **Input**: A must-watch for anyone.**Output**: Positive (1).
  4. **Input**: A waste of time and money.**Output**: Negative (0).
  5. **Input**: Heartwarming and delightful. → **Output**: Positive (1).
  6. **Input**: Avoid this at all costs. → **Output**: Negative (0).
  7. **Input**: An amazing performance. → **Output**: Positive (1).
  8. **Input**: Terrible in every way. → **Output**: Negative (0).
  9. **Input**: Beautifully directed. → **Output**: Positive (1).
  10. **Input**: It lacked any sense of fun. → **Output**: Negative (0).
- **Difficult Instances (10 Examples):**
  1. **Input**: Not bad at all. → **Output**: Positive (1).

Double-negative can confuse models.

2. **Input**: It could have been better. → **Output**: Negative (0).
   Requires understanding implied sentiment.
3. **Input**: An interesting mix of good and bad moments. → **Output**: Neutral/Positive (1).
   Mixed sentiment, challenging classification.
4. **Input**: The film tries hard but ultimately fails. → **Output**: Negative (0).
   Contrastive sentiment (positive attempt, negative result).
5. **Input**: Leaves the audience wanting more. → **Output**: Positive (1).
   Ambiguity around wanting more.
6. **Input**: Some scenes are great, others fall flat. → **Output**: Neutral/Negative (0).
   Mixed polarity within a single sentence.
7. **Input**: The acting was fine, but the plot was dull. → **Output**: Negative (0).
   Requires prioritization of the dominant sentiment.
8. **Input**: Not a movie I'd recommend. → **Output**: Negative (0).
   Implied negativity without explicit words.
9. **Input**: Better than most, but still not great. → **Output**: Neutral/Negative (0).
   Complicated sentiment requiring subtle understanding.
10. **Input**: I have mixed feelings about it. → **Output**: Neutral/Negative (0).
    Explicit mention of ambiguity.

Reflection on the Annotation Setup

- **Annotator Sample:**

  The original SST annotations were performed by crowd workers on Amazon Mechanical Turk.

  Workers were likely native or fluent English speakers but may have varied in linguistic or cultural expertise.

- **Annotation Guidelines:**

  Annotation guidelines provided by the dataset creators included examples of positive, negative, and neutral sentiment.

  While detailed, nuances like sarcasm or idiomatic expressions might not have been well-anchored.

- **Inter-Annotator Agreement:**

  We could not find any information about how Inter-Annotator Agreement was resolved. However, it is reported that if all the annotations are taken on a scale from 1 to 25, the average variance is 9.7238.

- **Conflict Resolution:**

  Conflicts were resolved using majority voting among annotators.

  Instances with strong disagreement may have been excluded or flagged.

- **Dataset Quality:**

  Strengths: Balanced dataset, diverse sentence structures, and real-world applicability.

  Weaknesses: Some examples are ambiguous, and mixed sentiment instances might lack consistent annotation.

- **Opinion on Quality:**

  The quality of the SST-2 dataset is generally high for binary sentiment analysis due to its balanced class distribution, diverse sentence structures, and real-world applicability, which make it a robust resource for training and evaluating models. However, the selection of instances includes some ambiguous examples and sentences with mixed sentiment, which could challenge both annotators and models. The annotations, while effective for most cases, may lack consistency in handling nuanced expressions, such as sarcasm or cultural idioms, and do not always provide clear guidelines for ambiguous or mixed-polarity cases. Enhancements, such as clearer annotation protocols for these edge cases or additional labels for nuanced sentiment, could further improve the dataset's overall utility and reliability.

- **Explanation of Categorization:**

  Easy Instances: Explicit sentiment, simple vocabulary, no ambiguity.

  Difficult Instances: Ambiguous sentiment, idiomatic expressions, mixed polarity, or implied meaning.

Deliverable 2: Finetune

Objective

In this deliverable we fine-tuned GPT-2 on the Stanford Sentiment Treebank (SST-2) dataset from the GLUE benchmark, which involves binary sentiment classification. Additionally, we compared the performance of:

1. A pre-trained GPT-2 model fine-tuned on SST-2. (4 fine-tuning variants)
2. A randomly initialized GPT-2 model fine-tuned on SST-2.

Performance Analysis:

- Comparison of the two models on validation/test metrics.
- Analysis of performance on 20 manually selected instances (10 easy and 10 difficult).
- Reflection on results and their implications.

Dataset

- **Dataset**: SST-2 (Stanford Sentiment Treebank, GLUE benchmark task).
- **Task**: Binary classification of sentences into positive or negative sentiment.

Model Fine-tuning

1. **Pre-trained GPT-2 Fine-tuning:**
   - **Head-Only Tuning**

     In the head-only tuning approach, we trained only the last classification layer. Since this layer is not pre-trained with the original model, it represents the minimum number of training parameters required without using methods like prompting. We have 1,536 trainable parameters. However, this also means the model is limited to learning only linear dependencies during fine-tuning. This limitation is reflected in the results, which, along with partial fine-tuning, are worse than those achieved through more extensive training methods.

     The major advantage of this method, however, is its cost-effectiveness. It took only about 42 minutes on the university's HPC cluster, using a single A100 graphics card and 4 GB of RAM, to calculate 11 epochs. This time could likely have been reduced further by precomputing the mapping of GPT-2 model inputs into the vector space and then conducting the training based on that. Considering that the results are still decent, this approach remains a viable option for use with large models and datasets.

   - **Partial Fine-Tuning**

     This method is similar to Head-Only Tuning, but with more learnable parameters unfrozen in the final layers. In addition to the classification layer, the last Transformer block and the last Layer Normalization are now trainable. As with Head-Only Tuning, the training time reported here is not fully representative of the total time consumption, as we could have precomputed more than we did in this training run. However, for this

experiment, we required 38 minutes to complete 8 epochs. The validation accuracy is slightly worse than that achieved with the Head-Only Tuning method. However, since machine learning is inherently subject to variability, we do not consider this difference to be statistically significant.

- **Full Fine-Tuning** (Updating all GPT-2 parameters + classifier):

  In this approach, we update all trainable parameters of the pre-trained GPT-2 model, including the added classification head. The model contains 124,440,576 trainable parameters, which allows it to leverage the full capacity of GPT-2 for adaptation to the downstream task.

  The model starts with an initial train accuracy of 68.75% for the first 10 batches. Over 6 epochs, the training process significantly improves both training and validation accuracy. By the final epoch:

  - Train Accuracy: 99.34%
  - Validation Accuracy: 94.00%
  - Test Accuracy: 89.79%

  The training process takes approximately 2741.62 seconds (1:15 hours), reflecting the computational expense of updating the entire model. It should also be noted that the time gap between the different model would have been bigger if we would have used a preprocessing step in the head finetuning and partial finetung as stated the the paragraphes of these Methods. Loss consistently decreases over epochs, indicating effective convergence. This strategy achieves high performance at the cost of longer training time and computational resources due to the large number of parameters being updated.

- **LoRA** (Introducing Low-Rank Adaptation layers to reduce parameter count):

  The LoRA (Low-Rank Adaptation) technique modifies the pre-trained GPT-2 model by introducing lightweight, low-rank layers into specific parts of the model. This drastically reduces the number of trainable parameters to just 1.77 million (about 1.4% of the total parameters because only LoRA layers and classifier parameters are updated, with most GPT-2 layers frozen).

  The model starts with moderate training accuracy and validation accuracy in the first epoch:

  - Train Accuracy (Epoch 1): 73.84%
  - Validation Accuracy (Epoch 1): 81.07%

  Over 6 epochs, the performance steadily improves:

  - Train Accuracy: 84.83%
  - Validation Accuracy: 84.83%
  - Test Accuracy: 84.82%

  The training process is significantly faster, taking approximately 905.94 seconds (25 minutes) just about 33% of the time needed for full fine-tuning. The losses stabilize after several epochs, indicating convergence. While the LoRA approach achieves slightly lower test accuracy compared to full fine-tuning, it provides a

computationally efficient alternative with substantially reduced training time and memory requirements, making it suitable for resource-constrained environments.

2. **Randomly Initialized GPT-2 Fine-tuning**

In this approach, the GPT-2 model starts with randomly initialized weights instead of leveraging pre-trained weights. A sequence classification head is added, and all parameters are updated during training. Since the model is not pre-trained, it must learn patterns and representations entirely from scratch, relying solely on the downstream task dataset.

The randomly initialized GPT-2 contains 124,440,576 trainable parameters, the same as the fully fine-tuned pre-trained model. However, the lack of pre-training results in significantly lower initial performance. Before training, the baseline accuracies were as follows:

- Train Accuracy: 53.12%
- Validation Accuracy: 45.62%
- Test Accuracy: 49.38%

These results indicate that the model performs close to random guessing for binary classification tasks due to its lack of meaningful learned patterns. After seven epochs of fine-tuning, the model achieved the following performance:

- Train Accuracy: 99.48%
- Validation Accuracy: 88.54%
- Test Accuracy: 78.67%

The model's failure to generalize can be attributed to starting with randomly initialized weights, requiring it to learn language patterns and task-specific details entirely from scratch, which makes it highly dependent on the quality and diversity of the dataset. Without pre-trained embeddings, the model struggles to understand semantic relationships and handle nuanced cases in the test data. Additionally, the relatively short training duration and limited computational resources, while achieving high training accuracy, may not allow the model to fully converge on a solution that generalizes well. Despite these limitations, this approach remains an option for tasks where pre-trained weights are unavailable or unsuitable.

3. **conclusion**

Through our exploration of different fine-tuning approaches for GPT-2, we've observed a clear trade-off between performance, efficiency, and adaptability. Full fine-tuning of the pre-trained model delivered the highest test accuracy (89.79%) by utilizing all parameters, but it required significant computational resources and time. In contrast, the LoRA approach offered a more practical solution, achieving solid results (84.82%) with a fraction of the parameters and expected training time (with proper optimization), making it an excellent choice for scenarios with limited resources. Head-Only and Partial Fine-Tuning methods were faster and more cost-effective but showed limitations in accuracy due to their restricted trainable parameters. The randomly initialized GPT-2 model shows the critical importance of pre-trained embeddings. While we achieved decent training accuracy after several epochs, the model struggled to generalize, with a lower test accuracy (78.67%). Each method presents its own strengths and weaknesses, and the choice

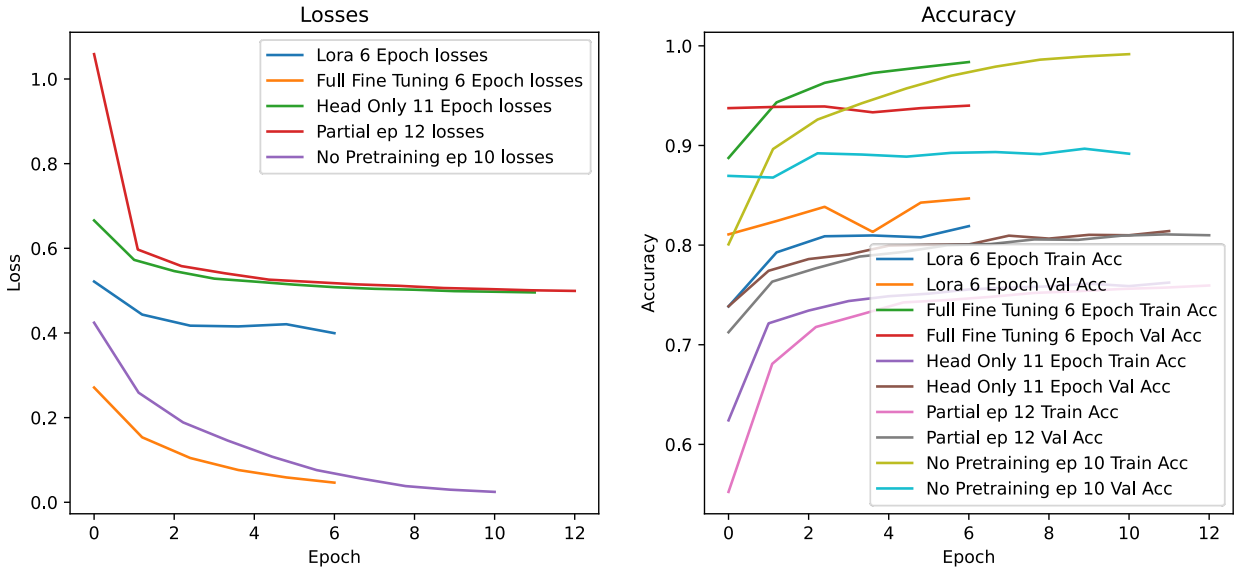
ultimately depends on the balance between resource constraints and performance needs for the given task.

Performance Analysis

- Comparison of the models based on validation/test metrics. (with plots)

  LoRA achieved a balanced performance, with high validation and test accuracies (84.83% and 84.82%) while keeping the training time relatively low compared to full fine-tuning. Full Fine-Tuning reached the highest validation accuracy (94.00%) and test accuracy (89.79%), but at a much higher computational cost (2741.62 seconds). Head-Only Tuning and Partial Fine-Tuning showed comparable results in terms of accuracy, though they were significantly faster to train.

Model	Epochs	Train Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)	Training Time (s)	Time per Epoch
LoRA	6	84.83	84.83	84.82	905.94	151
Full Fine-Tuning	6	99.34	94.00	89.79	2741.62	457
Head-Only Tuning	11	81.07	81.41	82.34	2372.4	215
Partial Fine-Tuning	12	81.00	80.99	81.65	1861.6	155
No Pre Training	10	99.79	89.17	78.44	2655.39	265



- Analysis of performance on 20 manually selected instances (10 easy and 10 difficult).

  Full Fine-Tuning consistently performed the best, correctly classifying all easy and difficult instances. LoRA struggled with difficult instances, suggesting that its parameter-efficient nature may limit its ability to capture nuances. Both Head-Only Tuning and Partial Fine-Tuning achieved comparable results, correctly classifying 9/10 difficult instances.

Model	Accuracy (Easy)	Accuracy (Difficult)
LoRA	90% (9/10)	70% (7/10)
Full Fine-Tuning	100% (10/10)	100% (10/10)
Head-Only Tuning	100% (10/10)	90% (9/10)
Partial Fine-Tuning	100% (10/10)	90% (9/10)
No Pre Training	90% (9/10)	30% (3/10)

- Reflection on results and their implications.
  - Trade-offs Between Efficiency and Accuracy: LoRA is highly efficient and delivers competitive accuracy for most tasks but struggles with nuanced or ambiguous inputs. This makes it suitable for applications with resource constraints.
  - Full Fine-Tuning: While achieving the best overall performance, its high computational cost makes it less practical for scenarios with limited resources.
  - Head-Only and Partial Fine-Tuning: These methods provide a good balance between efficiency and accuracy. They are particularly useful when computational resources are limited, yet a moderate level of accuracy is required.
  - LoRA could be used for fast and cost-effective deployments with general tasks. Full Fine-Tuning could be used for tasks requiring high precision or nuanced understanding. Opt for Head-Only Tuning or Partial Fine-Tuning could be used for a balance between performance and efficiency.
  - We were surprised by how significant the difference between the 2 categories were in no pre-tuning model. It shows once more how bad this model is at generalization.

Challenges

The main challenges were of a technical nature. Since some of the required libraries could not be installed, we had to create a custom container for the JupyterHub on the HPC. This process required extensive testing and learning, as none of us had prior experience with Docker. Additionally, different sources from the HPC used different parent containers in their examples, and some of these were incompatible with our packages. For some reason, we were also unable to install pip packages after the container was running, which meant we had to create a new container every time we needed a new package.

During the implementation phase, we encountered several minor errors that took more time to debug than anticipated. For example, we incorrectly added the padding token, which caused CUDA errors that we needed to investigate and resolve.

On the positive side, we adopted a more standardized and clear implementation approach from the beginning, which was partially compatible with our code from the previous week, allowing us to reuse some of it. This also made reruns of the code, whether to fix mistakes or to recreate results during the report writing phase, far less time-consuming than in the previous project. Additionally, we started saving runs in a format suitable for later plotting earlier in the process than we did in the prior project. However, this came at the cost of investing time into refactoring some of the code early on.

Deliverable 3: Multilingual Model

Option 1: Qualitative Evaluation

40 prompts in Persian that systematically vary by one of these characteristic (e.g., register, domain, syntactic complexity, tense, and other):

#	Sentence	Model Translation	Actual Translation	Category	Type
1	لطفاً پنجره را باز کنید.	Please open the window.	Please open the window.	Register	Formal
2	پنجره رو باز کن.	Open the window.	Open the window.	Register	Informal
3	آیا می‌توانید به من کمک کنید؟	How can I help you?	Are you able to help me?	Register	Formal
4	می‌تونی کمکم کنی؟	I can help you?	Can you help me?	Register	Informal
5	لطفاً دفتر خود را به من بدهید.	Please let me know your office.	Please give me your notebook.	Register	Formal
6	دفترتو بده.	I will give you my office.	Give me your notebook.	Register	Informal
7	آیا امکانش هست که چند دقیقه صبر کنید؟	Is it possible to wait for a few minutes?	Is it possible to wait for a few minutes?	Register	Formal
8	میشه یه لحظه صبر کنی؟	Can you wait for a moment?	Can you wait a moment?	Register	Informal
9	امروز هوا آفتابی است.	Today the weather is sunny.	It's sunny today.	Domain	General
10	سیستم عامل نیاز به بروزرسانی دارد.	The operating system needs to be updated.	The operating system needs to be updated.	Domain	Technical
11	من عاشق کتاب خواندن هستم.	I am a lover of reading.	I love reading books.	Domain	General
12	هیلوم نقطه‌ای است که در آن شریانهای حامل مواد مغذی و لنفوسیت‌ها وارد غده لنفاوی می‌شوند و سیاهرگ‌ها از غده لنفاوی خارج می‌شوند.	A point in which the blood vessels that carry nutrients and leukocytes enter the gland of the liver.	The hilum is the point where arteries carrying nutrients and lymphocytes enter the lymph node and veins exit the lymph node.	Domain	Technical (Medical)
13	یک فنجان قهوه لطفاً.	A cup of tea please.	A cup of coffee, please.	Domain	General
14	خیار فسخ اصطلاحی در فقه و حقوق به معنای حق برهم‌زدن یک‌جانبه قرارداد است.	Termination of a Contract	Option of termination is a term in jurisprudence and law that means the right to unilaterally cancel the contract.	Domain	Technical (Law)

15	من به خرید می‌روم.	I want to buy.	I'm going shopping.	Domain	General
16	انرژی جنبشی عبارت است از کار مورد نیاز برای شتاب دادن به جرم جسم برای رسیدن به سرعت مورد نظر از حالت سکون	Static energy refers to the required work to accelerate the body for reaching the desired speed.	Kinetic energy is the work required to accelerate the mass of an object to a desired speed from rest.	Domain	Technical (physics)
17	من به خانه رفتم.	I went to the house.	I went home.	Syntactic Complexity	Simple
18	بعد از این که کارم تمام شد، به خانه رفتم.	After I finished my work, I went to the house.	After I finished work, I went home.	Syntactic Complexity	Complex
19	او آن کتاب را خرید.	I bought the book.	He/She bought the book.	Syntactic Complexity	Simple
20	او کتابی را که دیروز دیدم خرید.	I bought a book yesterday.	He bought the book I saw yesterday.	Syntactic Complexity	Complex
21	ما فیلم دیدیم.	We watched the movie.	We watched the movie.	Syntactic Complexity	Simple
22	وقتی باران تمام شد، ما به تماشای فیلم رفتیم.	When the rain stopped, we went to see the movie.	When the rain stopped, we went to watch the movie.	Syntactic Complexity	Complex
23	او دوید.	I ride.	He/She ran.	Syntactic Complexity	Simple
24	او به سمت درختی که در دوردست بود دوید.	He went to the tree that was in the dirt.	He/She ran to a tree in the distance.	Syntactic Complexity	Complex
25	من امروز صبح ورزش کردم.	Today I was exercising.	I exercised this morning.	Tense	Past
26	من هر روز ورزش می‌کنم.	I exercise every day.	I exercise every day.	Tense	Present
27	فردا صبح ورزش خواهم کرد.	Tomorrow I will go to the gym.	I will exercise tomorrow morning.	Tense	Future
28	او کتابی خواند.	He read a book.	He/She read a book.	Tense	Past
29	او کتابی می‌خواند.	He reads a book.	He/She is reading a book.	Tense	Present
30	او کتابی خواهد خواند.	He will read a book.	He/She will read a book.	Tense	Future
31	ما به رستوران رفتیم.	We went to the restaurant.	We went to the restaurant.	Tense	Past

32	ما در رستوران هستیم.	We are in the restaurant.	We are at the restaurant.	Tense	Present
33	ما به رستوران خواهیم رفت.	We will go to the restaurant.	We will go to the restaurant.	Tense	Future
34	خیلی زرنگی شما!	No Result Found????	You're so smart!	Miscellaneous	Sarcasm
35	بخشش لازم نیست اعدامش کنید.	It is not necessary to punish the murderer.	Forgiveness is not needed, execute him./ Forgiveness is necessary, don't execute him.	Miscellaneous	Ambiguity (Punctuation)
36	و تو چون مصرع شعری زیبا، سطر برجسته‌ای از زندگی من هستی	I am a beautiful Persian verse, a line of life I have.	And like a beautiful verse of poem, you are the highlight of my life	Miscellaneous	Complex (Poem)
37	هرچند خسته بودم، تا آخر شب بیدار ماندم.	I was tired, until the end of the night.	Even though I was tired, I stayed up late.	Miscellaneous	Concessive
38	اگر برف نبارد، به پارک می‌رویم.	If snow falls, we go to the park.	If it doesn't snow, we will go to the park.	Miscellaneous	Conditional
39	کتابی که دیروز خریدی، کجاست؟	A book you bought yesterday, where is it?	Where is the book you bought yesterday?	Miscellaneous	Embedded clause
40	من صلاح نمی‌بینم که آنها حرف بزنند.	I don't know how they say.	I do not see it fit for them to talk.	Miscellaneous	Complex verb

Multilingual Analysis:

1. **Register**:
   The translations of numbers 1, 2, 7, and 8 are accurate and perfectly capture the intended meaning.
   In number 3, the model misunderstood the structure and meaning, converting a question asking for help into a question offering help.
   In number 4, the model misinterpreted the question and turned it into a confusing declarative sentence.
   In numbers 5 and 6, the model mistranslated "notebook" as "office," which entirely changes the meaning. Pronouns were also not translated correctly. However, it is interesting to note that the Persian word in the original text has two meanings ("notebook" and "office"), and the model incorrectly chose the latter.
2. **Domain**:

   The translations of numbers 9 and 10 are correct.

   The translation of number 11 is partially correct. While it captures the general idea, the phrasing is unnatural and awkward in English.

Number 12 is incorrect. The model translated “lymph node” as “gland of the liver,” which is a major error in a medical context. Additionally, the second part of the sentence was not translated at all, making the entire translation incomplete and incorrect.

Number 13 is also flawed. The model mistranslated “coffee” as “tea,” but aside from that, the rest of the translation is accurate.

The translation of number 14 was incomplete.

Number 15 was mistranslated. The model turned an activity (“going shopping”) into a desire (“I want to buy”), which changes the meaning entirely.

In number 16, the model confused “kinetic energy” with “static energy,” which completely changes the meaning. The overall translation is incorrect and nonsensical.

3. **Syntactic Complexity**:

   The model performed well on some examples of simple and complex syntax but failed on others.

   For instance, in numbers 19 and 20, the subject was changed from “he/she” to “I,” which is a strange and unnecessary alteration.

   In number 23, the model produced a completely unrelated translation that does not match the original sentence.

   In number 24, the model mistranslated “distance” as “dirt,” which is both semantically and contextually incorrect.

4. **Tense**:

   The model performed well in translating tenses overall. However, In number 25 and number 27, where adverbs of time like “today” and “tomorrow” are used, the model failed to produce grammatically correct English sentences. The result sounds like an attempt by a Persian speaker unable to distinguish proper English tense structures.

5. **Miscellaneous:**

   In number 34, the model failed to produce any translation for the sarcastic sentence, which is unexpected.

   In number 35, punctuation is shown to completely alter the meaning of the sentence. The model chose the interpretation based on one possible punctuation, but the ambiguity remains unresolved.

   In number 36, the model struggled with the poetic nature of the sentence, resulting in a translation that was irrelevant and lacked fluency.

   In number 37, the model failed to convey the contrast expressed in the original sentence.

   In number 38, the model translated the condition as the complete opposite of the original meaning.

   In number 39, while the model translated the sentence correctly, the phrasing was awkward and reflected a Persian sentence structure rather than natural English.

   In number 40, the model failed to produce a meaningful and correct translation.

Conclusion

Overall, the model does well with simple sentences and straightforward translations but struggles when faced with ambiguity, complex structures, or nuanced expressions like sarcasm and contrast. It also has trouble accurately translating technical or poetic content and sometimes produces unnatural sentence structures. These issues suggest the model needs better contextual understanding and more training to handle complex syntax and subtle linguistic differences effectively.

Deliverable 4: Project Summary

Key Findings:

As expected pre-trained models have the best performance compared to other models specially when dealing with complicated sentences. Between the pre-trained models it is more cost efficient to use head-only since there are far less parameters to learn and the performance compared to re-training the whole model is not that significant. We tried prompting mGPT in persian however, it seems like the model was not able to understand the sentence in persian and kept repeating the prompt or the input sentence to generate the output.

Challenges:

As described above, we faced numerous challenges in getting the technical components to work properly. Working with the HPC cluster and Docker was particularly challenging for us. We also encountered typical problems related to PyTorch, such as CUDA errors that were difficult to debug and mismatched tensor shapes. Additionally, working with Persian (right-to-left) and English (left-to-right) simultaneously proved to be quite challenging. For the non-Persian-speaking team member, it was especially complicated, as it was difficult to determine whether a copy process had corrupted the input. Furthermore, sending partial results via messaging was problematic when left-to-right and right-to-left text were combined on the same line.

Work Process:

Merle did deliverable 2 and most of the codings and Moujan and Farhan worked on deliverable 1 and 3. Moujan also worked on the report and Merle and Farhan went through the whole report again. We all tried to help each other out in the problems we faced (in coding or analysis of something). It is worth mentioning that we used ChatGPT for grammar check and some rephrasing of sentences in this report.