Central Asian University

Sentiment Classification of Uzbek E-commerce (Uzum Market) Customer Reviews

Main goal is to build and compare multiple **NLP Transformer** models
for sentiment classification of **Uzbek** reviews.

📌 Results of Transformer:

Transformer encoder model achieved a training and validation **loss of 0.28 and 0.32** on Uzbek review sentiment classification

Why Loss = 0.28 Is Meaningful:

- Cross-entropy loss **close to 0** means:
- predicted probability mass concentrated on correct class
- For 3-class classification:
- random **guess ≈ 1.10 loss**
- Transformer **model ≪ random baseline**

Compared to a random classifier, the Transformer **significantly reduces uncertainty in predictions.**

📌 Outputs from Model:

```
# testing 1
text = "oyimga ko'rsattim bo'lar ekan, yoqmasa kerak deb o'ylagandim"
probs = model.predict(text, tokenizer)[0].tolist()
out = np.argmax(probs)
print(probs)
print('postive' if out == 2 else 'neutral' if out == 1 else 'negative')
```

```
[0.14750970900058746, 0.06039566546678543, 0.7920945882797241]
postive
```

```
# testing 2
text2 = "oyimga ko'rsattim bo'lar ekan, yoqmasa kerak deb o'ylagandim, lekin o'zimga to'grisi vashe ishlashi yoqmadi"
probs2 = model.predict(text2, tokenizer)[0].tolist()
out2 = np.argmax(probs2)
print(probs2)
print('postive' if out2 == 2 else 'neutral' if out2 == 1 else 'negative')
```

```
[0.8540540933609009, 0.109282948076725, 0.036662884056568146]
negative
```

```
# testing 3
text3 = "telefon boshida yaxsh ishladi, keyin o'zidan o'zi ekrani ishlamiy qoldi"
probs3 = model.predict(text3, tokenizer)[0].tolist()
out3 = np.argmax(probs3)
print(probs3)
print('postive' if out3 == 2 else 'neutral' if out3 == 1 else 'negative')
```

```
[0.5298020839691162, 0.19840247929096222, 0.271795392036438]
negative
```

📌 Why Uzbek-language based model

Motivation:

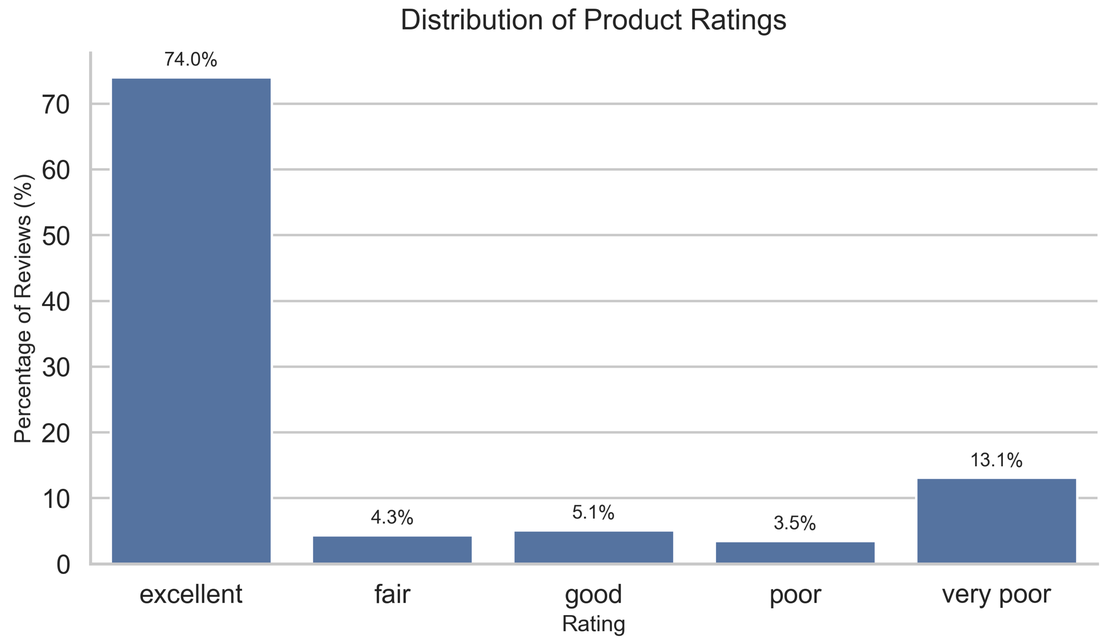
- Uzbek language is **low-resource** in NLP
- Real business value:
  - **automatic** customer satisfaction **monitoring**
  - **rating prediction**
  - feedback analysis for marketplaces (Uzum)

Challenges:

- noisy user-generated text
- mixed scripts (**Latin + Cyrillic)** and some **Russian mixed texts**
- non-standard characters

📌 Dataset Description:

- Source: **Uzum Market** Customer reviews
- Size: **~350k reviews**
- Class **imbalance**
- **Average** review length = **8 words**
- Language mixture (**Latin + Cyrillic+Russian**)



📌 Text Normalization:

This step significantly reduces **vocabulary size** and **noise**.

Why normalization is critical for Uzbek:

- different alphabets
- accented characters
- foreign symbols

What I did:

- Unicode-based filtering (\p{Latin}, \p{Cyrillic})
- Character-level normalization mapping
- Removal of unwanted symbols
- Preservation of spaces

```
Before: "Rahmat!!! Júdá yoqdi "
After:  "rahmat juda yoqdi"
```

📌 Baseline Models:

1. BoW + Linear Classifier

- CountVectorizer
- Linear layer
- Fast but ignores word order

2. BoW + Hidden Layers

- Token embeddings
- Increased number of neurons
- Simple semantic representation

3. Transformer Encoder Classifier

- Token embedding
- Positional embedding
- **4 Transformer blocks:**
  - Multi-head self-attention
  - Feed-forward network
  - Residual connections
  - LayerNorm
- Mean pooling over sequence
- Linear classifier head

📌 Model Complexity and Architecture:

Component	Trainable Parameters
Token Embedding Layer	3,840,000
Positional Embedding	4,096
Self-Attention Blocks	196,992
Feed-Forward Blocks	395,136
Classification Head	387
Total Parameters	**4,436,611**
Number of epochs	5
Learning rate	2×10^{-4}
Training time	~12 minutes on GPU

Thank You

Bakhtiyor Bekmurodov - 230035

Source code: https://github.com/baxt1or/ml-final-exam/blob/main/scripts/train_transformer.py

LinkedIn: https://www.linkedin.com/in/bakhtiyor-bekmurodov-farhod-ogli/

Central Asian University