Institute of Technology of Cambodia
Department of Applied Mathematics and Statistics

Khmer Text Prediction

TEAM 4

Lecturer: Mr. KHEAN Vesal

TEAM MEMBER

SUON Senchey

Sem Yuthearylyhour

Vey Sreypich

Vanna Juuka

Vorn Seavmey

TABLE OF CONTENT

Introduction & Motivation

Problem Definition & Challenges

Dataset & Data Collection

Data Preprocessing

Modeling Approaches Overview

Conclusion & Demo

Introduction & Motivation

I. Introduction & Motivation

What is Khmer Text Prediction?

- A system that predicts the next word
- Suggests possible words while typing in Khmer
- Used in keyboards, search bars, and messaging apps.

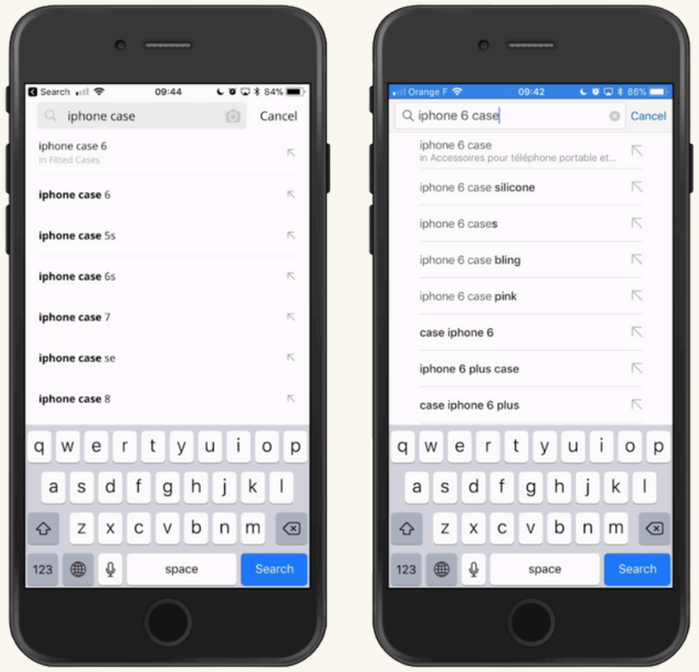
Why It Matters?

- Typing efficiency: fewer keystrokes, faster input.
- Accessibility: helps users with unlimited typing ability.
- Mobile-first language: improves everyday communication.

Existing Applications:

Khmer Smart Keyboard

Khmer eKeyboard



Problem Definition & Challenges

II. Problem Definition & Challenges

Problem Definition

- **Task:** Predict the next word given previous text.
- Formulated as a language modeling problem.

Example:
Input: ខ្ញុំចង់ទៅ
Output: សាលារៀន

Khmer-Specific Challenges

a. No Clear Word Spacing

- Khmer text does not separate words with spaces
- Hard to identify word boundaries

Example: ខ្ញុំចង់ទៅសាលារៀន។

b. Semantic Ambiguity

- Multiple words combine into one semantic unit
- Increases ambiguity in prediction.

Example: សាលារៀន.

c. Orthographic Variants

- Same word can be written in different valid form.

Example: នារី & ស្ត្រី

Literature Review

Research Paper	Core Methodology	Key Findings	Direct Relevance to Khmer Project
Li et al. (2025) *Data-Centric Interpretability*	**Representer Theorem**: Analyzes "Support Samples" (Attract vs. Deter) influencing predictions.	Support samples are often **semantically rich tokens** (verbs/nouns); easy samples prevent overfitting.	**Data Strategy**: Khmer corpus must prioritize diverse, high-content samples over simple repetitive sentences.
Phan et al. (2023) *mT5 Cross-Lingual Transfer*	**Zero-Shot Evaluation**: Performance testing across 50+ languages (QA, Generation).	Performance is dictated by **pre-training volume** and **script similarity** (Latin vs. others).	**The Central Challenge**: Khmer's unique script and morphology make standard mT5 sub-optimal; requires script-aware tuning.
Arora et al. (2020) *Memorization vs. Generalization*	**Theoretical Analysis**: Comparing N-gram (Statistical) vs. RNN (Neural) models.	N-grams **memorize** sequences; Neural models **generalize** via latent linguistic structures (syntax).	**Model Selection**: Provides rationale for Neural models to handle Khmer orthographic variants that N-grams cannot.

Dataset & Data Collection

Khmer Text Sources

To capture real-world typing patterns, we collect text from multiple domains:

a) News & Social Media

b) NLP Project

c) Open Source (hugging face): Credit: kimleang123

Dataset Size & Format

- Dataset Size

+ 276,934 Khmer sentences
+ 363379 Tokens

60%

News & Social Media

15%

NLP Project

5%

Open Source

- Data Format

+ Plain text (.txt)
+ One sentence per line

```
1  សាលារាជធានីមិនទាន់ទទួលបានលិខិតសុំធ្វើបាតុកម្មរបស់ក្រុមសម្ព័ន្ធយុវជនគាំទ្រគណបក្សប្រជាជននៅឡើយ
2  ភ្នំពេញសាលារាជធានីភ្នំពេញនៅមិនទាន់ទទួលបានលិខិតជាផ្លូវការណាមួយដើម្បីស្នើធ្វើបាតុកម្មពីក្រុមសម្ព័ន្ធយុវជន
3  នេះបើតាមលោកឡងឌីម៉ង់អ្នកនាំពាក្យសាលារាជធានីភ្នំពេញបានថ្លែងប្រាប់នៅព្រឹកថ្ងៃចន្ទនេះ
4  យប់នេះប៉ូលិសដាក់បុស្តិ៍រហូតដល់៧កន្លែងបងប្អូនស្រីងស្រាសូមកុំបើកបរជៀសវាងការពិន័យធ្ងន់ធ្ងរនិងគ្រោះថ្នាក់
5  សមត្ថកិច្ចនឹងដាក់បុស្តិ៍ត្រួតពិនិត្យចាប់ពីវេលាម៉ោង៩យប់ដល់ម៉ោង៣ទៀបភ្លឺ
```

Data Preprocessing

1. Text Cleaning

- Remove noise and irrelevant symbols.
- Normalize punctuation and spacing.
- Remove HTML tags, URLs,email addresses, emoji
- Unicode normalization (NFC)
- Lowercasing for mixed Khmer + English text.

2. Tokenization Strategy

- Use Lyhour Segmentation model (Ex: NGram)
- Tokenization: SentencePiece (Hugging face)

3. Sentence Segmentation

- Split text into one sentence per training sample
- Use Khmer punctuation (។ ? !)

Processing Steps	NGram	BiLSTM/LSTM/GRU	Transformer(GoldFish)
Tokenization Unit	Word-level	Subword	Subword
Sequence Length	Short (n fixed)	Medium (20–50 tokens)	Long (128Tokens)

Modeling Approaches Overview

Statistical vs Deep Learning Methods

- **Statistical models:** rely on word frequency & probability.
- **Deep learning models:** learn contextual representations from data.
- **Goal:** predict the next word given previous context.
- **Used Models:**
  + NGram: BiGram & TriGram.
  + RNN-based: BiLstm, Lstm and GRU.
  + Transformer based: GoldFish (khm_khmr_5mb).
- **Expectation:**

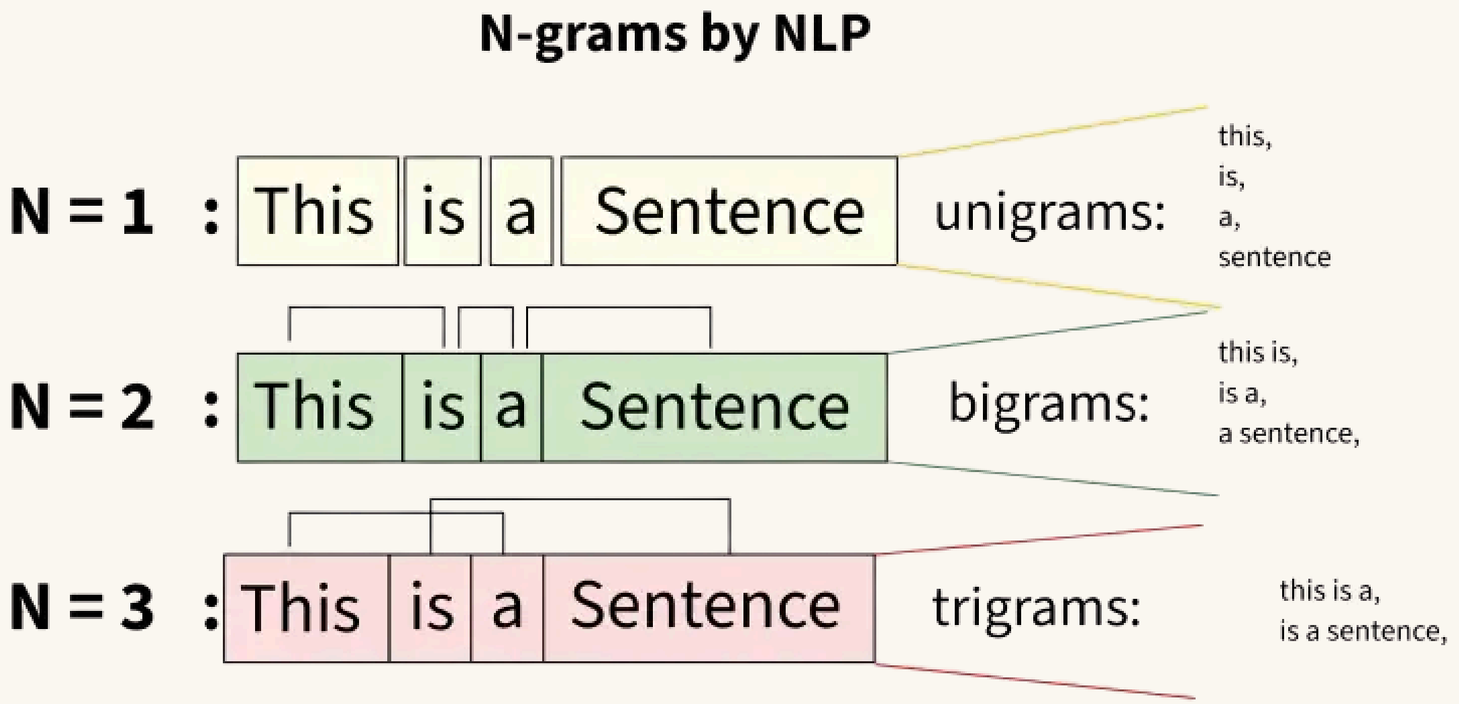
Model	Accuracy	Context length	Training Speed
N-gram	Low	Handle only short phrases	Very Fast
RNN-based	Medium	Short–medium sentences	Medium
Transformer-based	High	Designed for next-token prediction	Slow

NGram

- **Concept:**

+ Predicts the next word using the previous n−1 words.
+ Based on conditional probability.

Example: Input: ខ្ញុំចង់
Prediction: ទៅ



GRU

- **Concept**

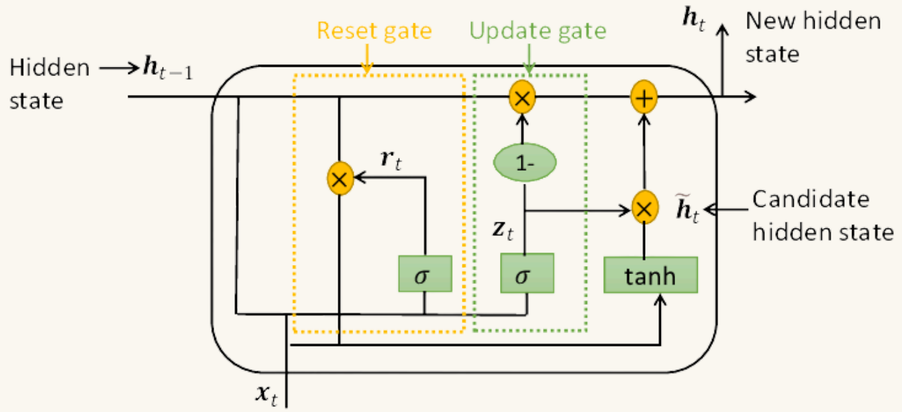
+ GRU is a recurrent neural network (RNN) designed to model sequential data.
+ Decide what to remember and what to forget from previous context.
+ Compared to LSTM, GRU has a simpler structure and fewer parameters.

- **GRU Gates Explanation**

+ Update Gate (z_t)
→ controls how much past information is kept.
+ Reset Gate (r_t)
→ controls how much past information is ignored.

- **Project Configuration**

```
MAX_LEN = 50
BATCH_SIZE = 32
EMBED_DIM = 128
HIDDEN_DIM = 256
NUM_LAYERS = 1
LEARNING_RATE = 1e-3
EPOCHS = 30
```



- **Architecture Overview**

1. Input at time step t : x_t
2. Previous hidden state: h_{t-1}
3. Gates compute new hidden state: h_t
4. Hidden state flows to the next time step

LSTM

- **Concept**

+ LSTM is a type of Recurrent Neural Network (RNN) designed to model long-term dependencies in sequential data.
+ It introduces a memory cell that can preserve information over long sequences.
+ Specifically designed to solve the vanishing gradient problem in standard RNNs.

- **Key idea:**

+ LSTM learns what to remember, what to forget, and what to output at each time step.

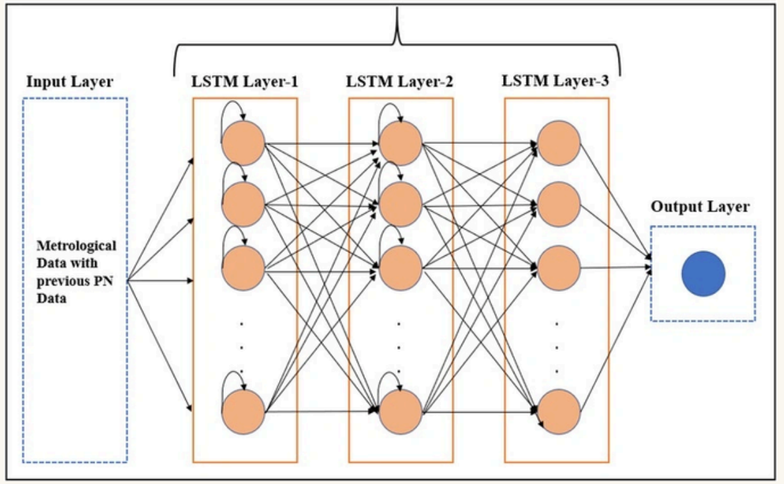
- **Example Dataflow**

Input sequence: ខ្ញុំ = x1, ចង់ = x2, ទៅ = x3

Step-by-step:

1. x_1 = ខ្ញុំ → stored in memory cell.
2. x_2 = ចង់ → LSTM keeps intent.
3. x_3 = ទៅ → combines past context.

Prediction: សាលារៀន



- **Project Configuration**

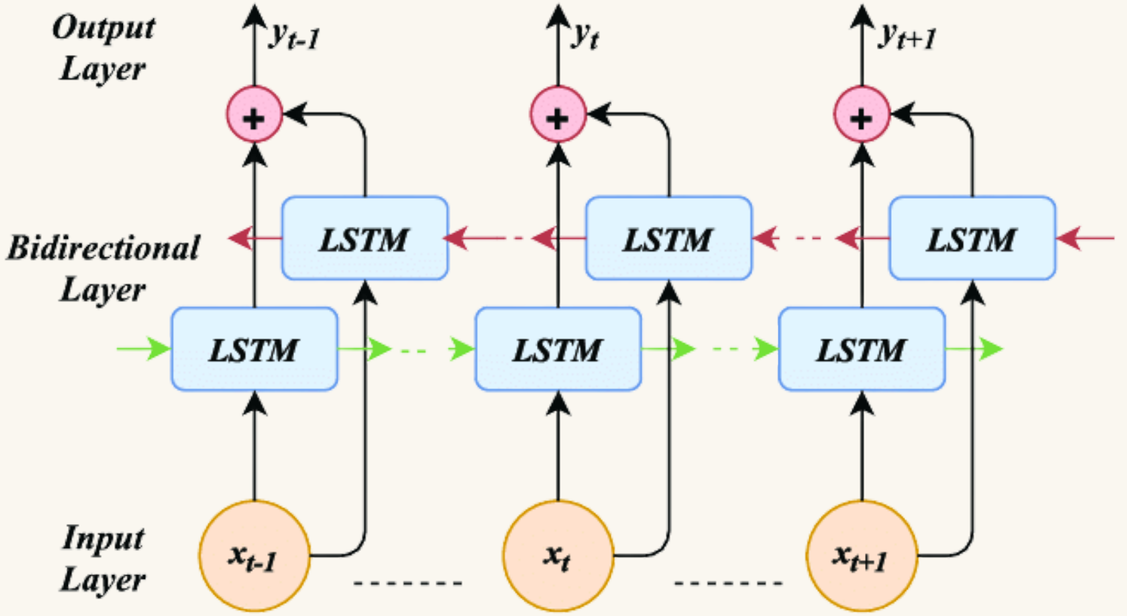
```
MAX_SEQ_LEN = 60
STRIDE = 20
BATCH_SIZE = 64
EPOCHS = 50
TOP_K = 5
EMBED_SIZE = 256
HIDDEN_SIZE = 512
NUM_LAYERS = 2
LR = 0.001
```

BiLSTM

- **Concept**

+ BiLSTM is an extension of LSTM that processes a sequence in two directions:
    - Forward (left → right)
    - Backward (right → left)

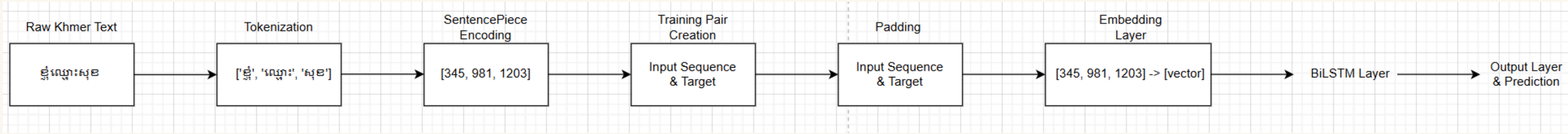
+ It combines past and future context for each word



- **Key idea:**

+ Understand a word using what comes before and what comes after.

- **Dataflow:**



Transformer based - GoldFish

Overview

- Architecture: gpt2.
- Parameters: 39M.
- Maximum sequence length: 512 tokens.

Citation

```
@article{chang-etal-2024-goldfish,
  title={Goldfish: Monolingual Language Models for 350 Languages},
  author={Chang, Tyler A. and Arnett, Catherine and Tu, Zhuowen and Bergen, Benjar
  journal={Preprint},
  year={2024},
  url={https://www.arxiv.org/abs/2408.10441},
}
```

Experiment: 2 Approach

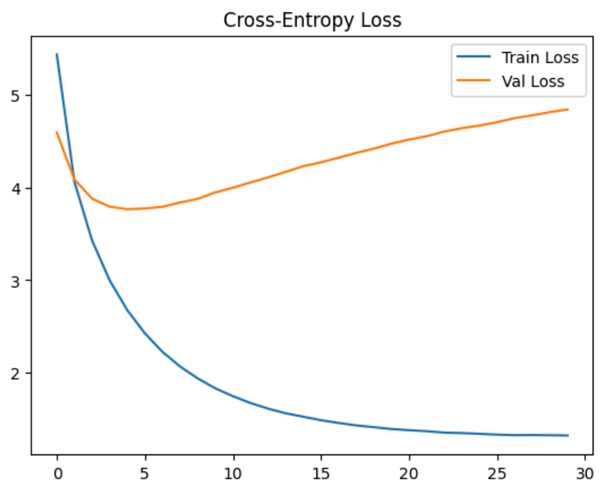
Model Comparison

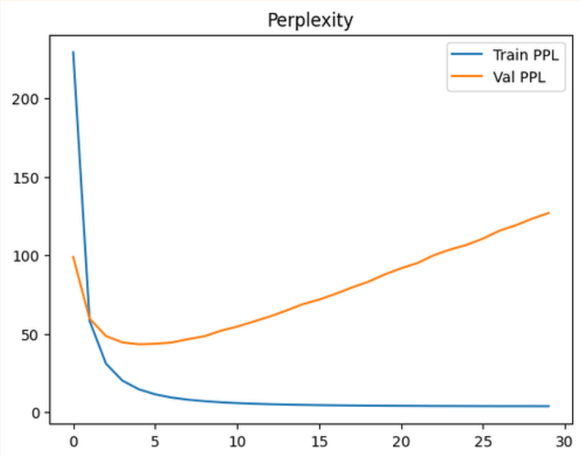
NGram

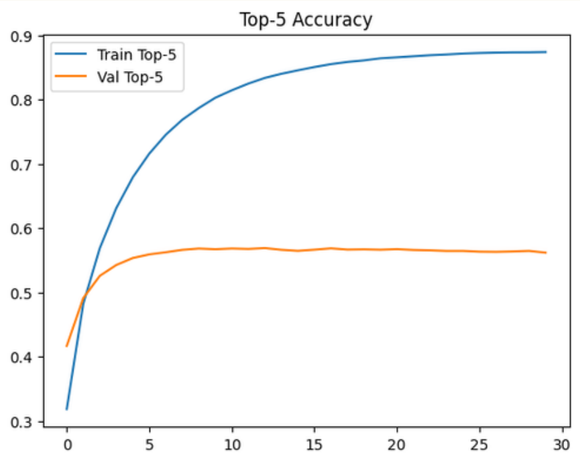
Testset	Bigram	Trigram	Measure
Loss	6.0882	8.5143	Lower
PPL	440.66	4985.67	Lower
Top-5	0.4099	0.5142	Higher[0-1]

Model Comparison

GRU







Train

- Loss: 1.3799
- PPL: 3.97

Val

- Loss: 4.5174
- PPL: 91.59
- Top-5: 0.5674

Test

- Loss: 4.7684
- PPL: 117.73
- Top-5: 0.5696

Model Comparison

LSTM

Train

- Loss 0.0671
- PPL 1.07

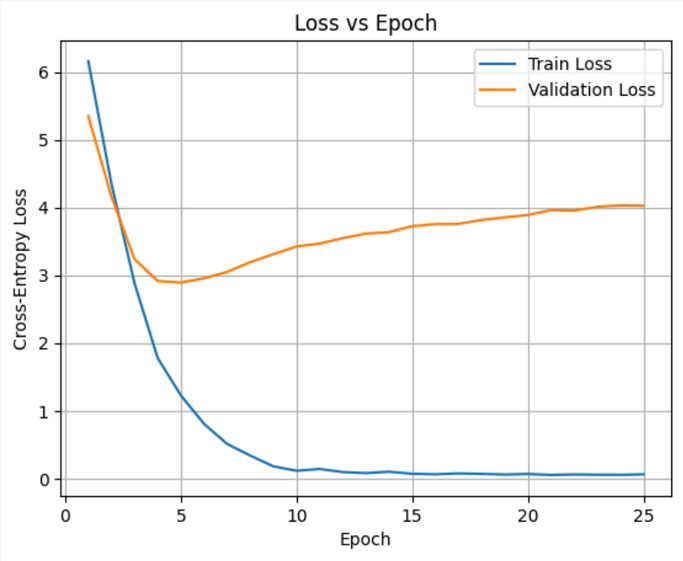
Val

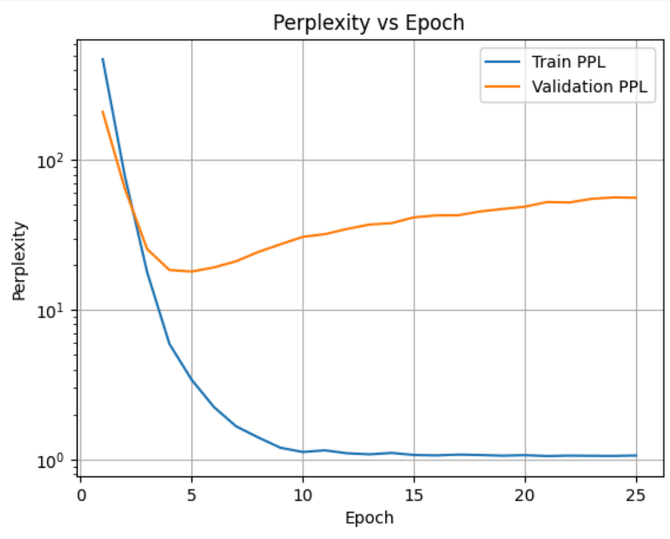
- Loss 3.6058
- PPL 36.81
- Val Top-5 0.7092

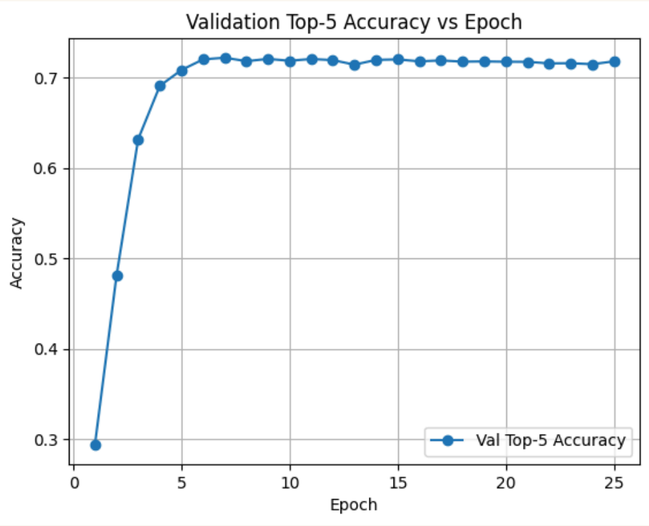
Test

- Loss 5.78
- PPL 45.56
- Val Top-5 0.651

BiLSTM







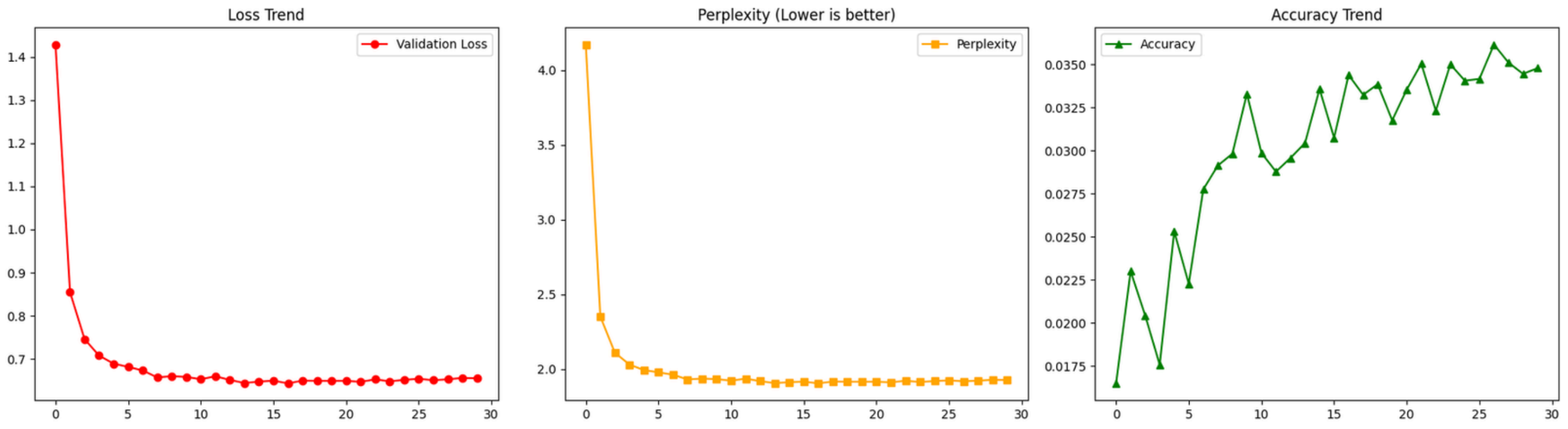
Model Comparison

GoldFish (SreyPich): Small dataset

Pretrained Model: goldfish-models/khm_khmr_5mb

Training Details:
Epochs: 30
Batch size: 8
Learning rate: 5e-5
Metrics: Loss, Perplexity, Accuracy

Training examples: 29,218
Validation examples: 7,305



Model Comparison

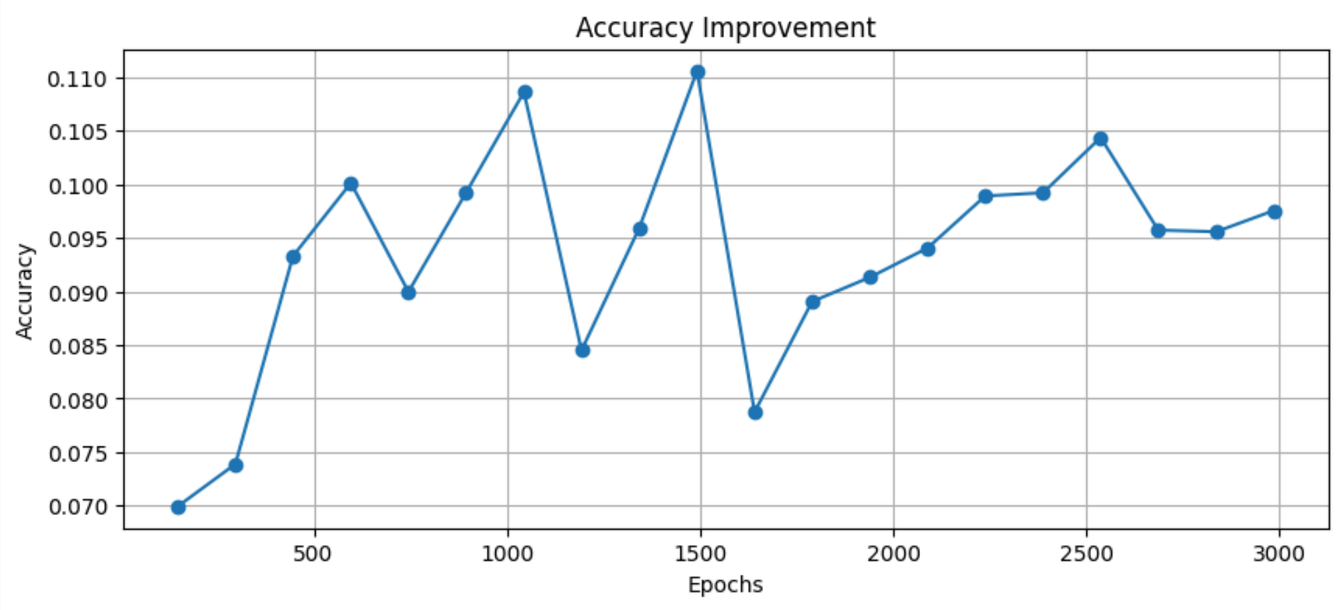
GoldFish (SreyPich): Small dataset

Pretrained Model: goldfish-models/khm_khmr_5mb

Training Details

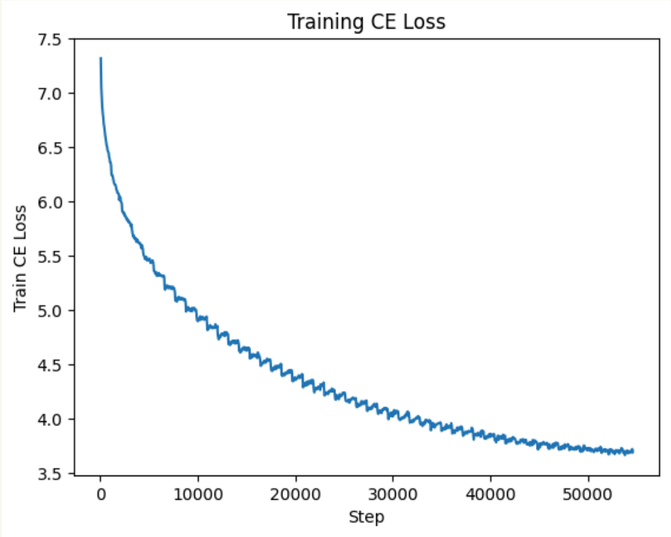
- Epochs: 20
- Batch size: 8 (train), 4 (validation)
- Learning rate: 5e-5
- Optimizer: AdamW (adamw_torch)
- Accuracy

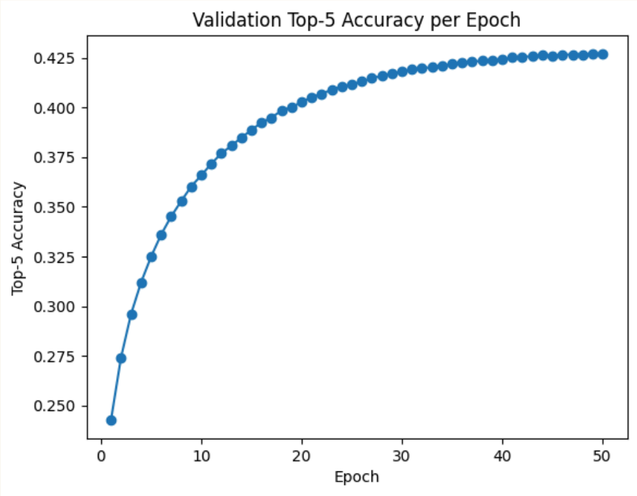
- Training set processed: 59,324 examples
- Validation set processed: 6,592 examples

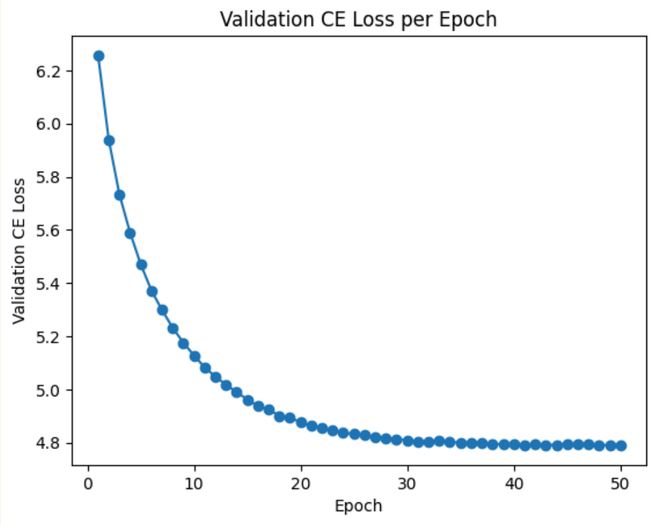


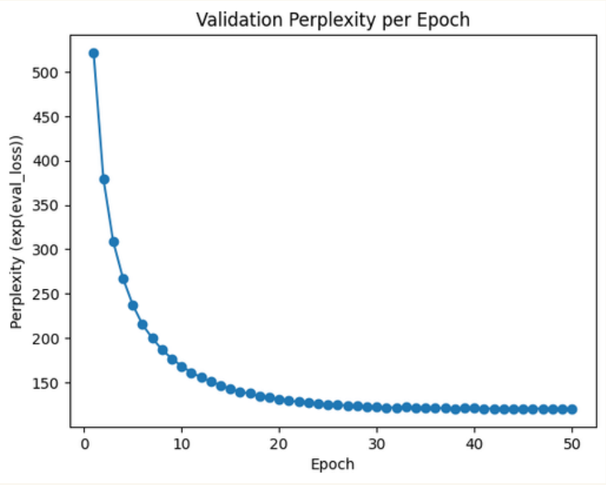
Model Comparison

GoldFish









Overall, BiLSTM achieved the best Performance.

Conclusion & Demo

Key Inspection and Summary

- **Initial Expectation ≠ Observed Result**
- **Dataset Size Was Small (Main Reason)**
- **Key point to say**
- With a small dataset, simpler models generalize better than large-capacity models.

Demo Specification

User → Web UI → /suggest API → BiLSTM model → Top-5 suggestions

TEAM 4

Thank You!

ANY QUESTIONS ARE WELCOMED.