KISS

Korean Intelligent Spacing Solver

Joonhyuk Cha, Kyu Yeon Lee, Jungwoo Park

Introduction

Korean is an agglutinative language where word spacing is crucial for readability and comprehension. Unlike languages with explicit word delimiters, Korean relies on spacing conventions that are often ambiguous and very complex, leading to frequent spacing errors in both manual and automated text generation. Deep learning techniques, particularly Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and Transformer-based models, have demonstrated significant potential in natural language processing tasks, making them strong candidates for solving Korean word spacing.

Problem Statement

Traditional approaches, such as rule-based algorithms, often fail in complex linguistic contexts and are infeasible to develop. While deep learning has shown promise, it is worth exploring the optimal architecture for this problem. The key challenge is to develop a model that effectively captures both local character-level dependencies and long-range contextual relationships, if necessary. This project aims to systematically compare CNNs, LSTMs, and Transformers using either the **Sejong Corpus** or the **2023 Newspaper Corpus** to determine the most effective architecture for Korean word spacing.

Objectives

1. Preprocess and utilize a high-quality corpus (Sejong or Newspaper Corpus 2023) for training and evaluation.
2. Develop and benchmark CNN, LSTM, and Transformer-based models for Korean word spacing.
3. Analyze the performance of each model in terms of accuracy, generalizability, and computational efficiency.
4. Determine the optimal approach for improving Korean word spacing, with a focus on practical deployment in real-world applications.

Methodology

- Dataset collection
    - To train and evaluate our models effectively, we will utilize a high-quality corpus that reflects real-world usage of Korean text. We plan to use either the Sejong Corpus or the 2023 Newspaper Corpus. These provide large-scale annotated text data with correct spacing, ensuring a reliable benchmark for our models.
- Data preprocessing
    - Text Cleaning – Removing unnecessary symbols, normalizing punctuation, and standardizing spacing inconsistencies.
    - Tokenization – Splitting text into characters and labeling them with binary indicators as either space or no space.
    - Encoding – Converting characters into numerical representations using word embeddings such as FastText or Byte Pair Encoding (BPE).
    - Data Splitting – Dividing the dataset into training, validation, and test sets while ensuring balanced representation of different spacing patterns.
- Model selection
    - We will implement and compare three deep learning architectures: LSTM, Transformer, and CNN. Each model will be evaluated based on its ability to learn local and global dependencies in text.
    - For LSTM, we will implement a bidirectional LSTM model to capture both forward and backward dependencies, use dropout and layer normalization to prevent overfitting, and train using cross-entropy loss and Adam optimizer.
    - For Transformer-based models we will implement a BERT-like transformer model fine-tuned on our dataset and apply multi-head attention and positional encodings to enhance contextual learning.
    - For CNN, we will use 1D convolutions with varying kernel sizes to detect character n-gram dependencies, stack multiple convolutional layers with batch normalization and ReLU activations, and incorporate a fully connected output layer for binary classification.

Expected Results and Discussion

As the task can be framed as a classification task (predicting a space or no space for each character in the input), standard classification metrics such as accuracy, precision, recall, and F1 score can be applied for evaluation. These metrics will allow us to compare the performance of the models and identify the types of errors that each model is prone to making.

We expect all of our models to achieve satisfactory performance on the task. Ultimately, our goal is to incorporate our trained models into useful tools for writing Korean, as Korean spacing is often tricky, even for native speakers.

References

[1] H.-C. KWON, M.-Y. KANG, and S.-J. CHOI, “Stochastic Korean Word-Spacing with Smoothing Using Korean Spelling Checker,” *International Journal of Computer Processing Of Languages*, vol. 17, no. 04, pp. 239–252, Dec. 2004, doi: https://doi.org/10.1142/s0219427904001103.
[2] J.-M. Choi, J.-D. Kim, C.-Y. Park, and Y.-S. Kim, “Automatic Word Spacing of Korean Using Syllable and Morpheme,” *Applied Sciences*, vol. 11, no. 2, pp. 626–626, Jan. 2021, doi: https://doi.org/10.3390/app11020626.
[3] S. Kim, G. Choi, and H. Kim, “Reliable automatic word spacing using a space insertion and correction model based on neural networks in Korean,” *Information Processing & Management*, vol. 56, no. 3, pp. 1046–1052, 2019, doi: https://doi.org/10.1016/j.ipm.2019.02.015.