Subword Unit Duration Modeling

Abstract

This project aims to model the duration of subword units (phonemes) in speech using time-aligned ASR data. We developed a baseline system that computes the mean duration for each phone from native English speakers and uses these means as predictions. And used multiple regression models. The implementation is designed in a modular, object-oriented Python codebase that processes data recursively from nested directories, evaluates predictions using standard metrics (MAE, MSE, R^2), and generates visualizations. This report details the methodology, implementation, evaluation, and future enhancements for subword duration modeling.

1. Introduction

Problem Statement

The goal is to predict phoneme durations using ASR alignments. Training is done on native English speakers; evaluation includes both native and non-native data to highlight temporal differences in pronunciation.

Significance

- Fluency Assessment: The model can help determine how closely L2 (non-native) speakers mimic native temporal speech patterns.
- Speech Synthesis: Accurate duration modeling is a key component in building natural-sounding TTS systems.
- Pronunciation Analysis: Detailed duration metrics can inform accent modification or pronunciation training systems.

2. Data Description

Data Sources

The project uses two primary datasets:

- Native Speakers: A subset of VoxForge data with self-reported native American English speakers.
- Non-Native Speakers: A subset comprising non-native speakers with varying degrees of English proficiency.

3. Methodology

Key Decisions Made

- **Feature Selection**: Initially, we opted to use only the encoded phone ID for simplicity. This decision was based on the assumption that the temporal behavior of each phoneme could be largely independent of context. However, this choice limited the model's ability to capture more complex, context-dependent variations in duration.

- **Model Selection**: We chose six regression models, balancing simplicity (Linear Regression) with more complex models (Random Forest and Gradient Boosting) to compare performance and see if more advanced methods would provide significant improvement over the baseline.

The baseline approach computes the mean duration for each phoneme using native speaker data:

- Training Phase: For every phone in the native training set, calculate the mean duration.

- Prediction Phase: For each phoneme in the test set (or non-native data), the predicted duration is set to the computed mean. For unseen phones, the overall mean is used as a fallback.

The other approach is the Regression. Six regressors from scikit-learn were trained using only the encoded phone ID as a feature:

- Linear Regression
- Decision Tree
- Random Forest
- Gradient Boosting
- HistGradientBoosting
- K-Nearest Neighbors

Each model was evaluated using:

- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)
- R^2 Score

4. Implementation

Code Structure & Modularity

The project is implemented in Python and organized into a modular, object-oriented codebase:

- config.py: Contains configuration constants (directory paths, output file paths, etc.).

- input_handler.py: Provides methods to load and parse JSON files recursively, extracting phoneme data.

- utilities.py: Includes helper functions (e.g., computing accuracy) and a DurationModeler class that handles training multiple models and evaluating predictions.

- output_handler.py: Manages writing results (CSV files, visualizations) to disk.

- main.py: The entry point that orchestrates data loading, model training, evaluation, and non-native predictions.

This structure ensures separation of concerns, adherence to PEP8 and Google style guidelines, and makes the code maintainable.

Data Loading & Preprocessing

1. Recursive File Search:
   The code uses:

   ```
   pattern = os.path.join(local_dir, '**', '*.json')
   json_files = glob.glob(pattern, recursive=True)
   ```

This searches for all JSON files within nested subdirectories.

Parsing JSON: The function `extract_phoneme_data` navigates the nested JSON structure (segments → words → phones) and extracts the phone and duration values into dictionaries.

Phone Encoding: To use phone labels as features, a mapping from phone strings to integer IDs is created by combining data from both native and non-native datasets.

Baseline & Model Training

- **Baseline Method**: Instead of sophisticated regression, the baseline computes the per-phone mean duration from training data. Predictions for the test set are made by looking up the phone's mean, with an overall mean fallback.

- **Regression Models:** In the extended implementation, multiple models (six regressors) are trained on the native data using only the phone_encoded feature to predict duration. Each model's performance is evaluated using MAE, MSE, and R².

Evaluation

- **Row-Level Predictions:** For each test record, predictions, absolute errors, and an accuracy measure are stored.
- Phone-Level Aggregation:

  The predictions are aggregated by phone to compute statistics such as:

  - Count per phone.
  - Average actual duration.
  - Average predicted duration per model.
  - Mean absolute error per model.
  - Average accuracy per model.
- **Overall Metrics**: Global performance metrics (MAE, MSE, R²) are computed for each model.
- **Visualizations**: Scatter plots and bar charts comparing actual vs. predicted durations, evaluating model performance visually.
- **Non-Native Predictions**: After training on native data, the models are applied to non-native data. If ground truth durations are available, error and accuracy metrics are computed; otherwise, only predictions are generated.

5. Results & Analysis

Baseline Results

- **Per-Phone Means**: For example, "SIL" (silence) has an average duration of ~0.3070 seconds, while consonants and vowels show different averages (e.g., "d" ≈ 0.0617 seconds, "a" ≈ 0.0816 seconds).
- **Test Predictions**: The baseline prediction for each test phoneme is directly the mean duration of its corresponding phone.

Regression Model Comparison

- **Performance Metrics**: For each regression model, metrics such as MAE, MSE, and R^2 are computed on the native test split. High R^2 values (e.g., above 0.80) indicate that some models (like RandomForest and GradientBoosting) explain a significant portion of the variance.

- **Row- and Phone-Level Analysis**: Detailed results at the row level and aggregated by phone provide insights into which phones or contexts are challenging for the models.

Non-Native Evaluation

- **Prediction Comparison:** Applying the native-trained models to non-native data allows us to quantify differences. For instance, if non-native durations for certain phones consistently differ from the native mean, that discrepancy can be used as an indicator of accent or fluency differences.

6. Discussion

Strengths of the Implementation

- **Modularity & Maintainability**: The refactored object-oriented code is well organized into modules, making it easy to extend and maintain.

- **Multiple Evaluation Levels**: The system supports row-level, phone-level, and overall metrics, providing a comprehensive picture of model performance.

- **Flexible Data Handling**: The recursive data loader easily manages nested subdirectories, ensuring all available data is processed.

Challenges Faced & Solutions

- **Data Imbalance**: Some phones are underrepresented in the dataset, which could negatively affect model performance. While we didn't implement advanced balancing techniques, we ensured that models were evaluated on both common and rare phonemes.

- **Non-Native Data Ground Truth**: Inconsistent or missing ground truth duration data for non-native speakers posed challenges for evaluation. We focused on providing predictions when ground truth was absent, but further refinement of the dataset or annotation process could improve error metrics.
- **Time Constraints**: One of the major challenges faced was the **time constraint**. Due to the limited time available for model development, we could not explore more complex approaches like **Neural Networks**. Neural network-based models, such as RNNs, LSTMs, or Transformers, have the potential to capture sequential dependencies and richer contextual features,

but training them would require significantly more time and computational resources. Given the project's scope and timeline, we opted to focus on regression models that could provide meaningful results within the available time frame.

Limitations

- **Feature Simplicity**: The current baseline and regression approaches rely solely on phone_encoded. This one-dimensional feature set may not capture complex context-dependent variations in duration.

- **Data Scale and Variability**: With over a million phoneme records, data imbalance (e.g., rare phones) may affect performance.

7. Future Enhancements

Contextual Feature Expansion

- Incorporate Additional Features:
  Include neighboring phones, syllable stress, word boundaries, and speaker-specific information to improve duration predictions.

- Prosodic Information:
  Integrate pitch, intensity, or speech rate as contextual features.

Advanced Modeling Techniques

Although the baseline is simple and interpretable, other approaches may offer improvements:

- Clustering Approaches:
  - Decision-Tree Context Clustering: Based on HMM-based TTS, this method groups phone-context states by asking linguistic questions.
  - Mixture Models / MDNs: Use Gaussian mixtures or neural mixture density networks to model multi-modal duration distributions.
- Neural Network Models:
  - Using RNNs, LSTMs, or Transformers to capture sequential dependencies and richer contextual information.

These alternatives can be compared against the baseline to determine the trade-off between simplicity and accuracy.

Improved Evaluation & Adaptation

- **Cross-Validation:** Apply k-fold cross-validation to ensure the stability of the baseline and regression models.

- **Speaker Adaptation:** Investigate models that can adapt to individual speaker characteristics, particularly for non-native speakers.

- **Real-Time Feedback:** Incorporate the model into an interactive system for language learning or accent training, where predictions guide real-time feedback.

8. Conclusion

This project successfully implements a system for predicting subword durations using native speaker data, with baseline methods and advanced regression models evaluated at multiple levels. The modular, object-oriented design supports future enhancements, including richer contextual features, clustering techniques, and advanced neural architectures. The resulting system provides a robust baseline for further research and real-world applications in speech fluency assessment and speech synthesis.

References

1. Rabiner, L. and Juang, B.H., 1993. *Fundamentals of speech recognition*. Prentice-Hall, Inc..

2. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, *12*, pp.2825-2830.

3. Bishop, C.M. and Nasrabadi, N.M., 2006. *Pattern recognition and machine learning* (Vol. 4, No. 4, p. 738). New York: springer.