Reduce and Reconstruct: ASR for Low-Resource Phonetic Languages

Anuj Diwan, Preethi Jyothi
Department of Computer Science and Engineering, Indian Institute of Technology Bombay, India
Introduction

● A seemingly simple but effective technique to improve E2E ASR systems for low-resource phonetic languages.
● E2E ASR is an attractive choice since speech is mapped directly to graphemes or subword units derived from graphemes.
● However, it is also very data-intensive and tends to underperform on low resource languages.
Introduction

● In our approach, we train two modules:
  a. an ASR system with a linguistically-motivated reduced output alphabet. For the ASR model, it is easier to learn and less data-intensive. (reduce)
  b. an FST-based reconstructor that recovers sequences in the original alphabet. (reconstruct)
● We run experiments on two Indian languages, Gujarati and Telugu.
● With access to only 10 hrs of speech data, we obtain relative WER reductions of up to 7% compared to systems that do not use any reduction.
Our Approach

1. **Devise a reduced vocabulary** that merges acoustically confusable and linguistically discriminative graphemes.

![Diagram showing relationships between Gujarati, Telugu, and IPA symbols]
Our Approach

2. Given labelled speech data, **transform transcriptions** using the reduction.
3. **Train** an **ASR system** that maps the original speech to the reduced transcriptions.

Sound wave saying ༄ slag
Our Approach

4. **Train a reconstructor** to reconstruct the original grapheme sequence from the reduced grapheme sequence.
Our Approach: FST-based Reconstructor

- **Input:** reduced-grapheme hypothesis from ASR system.
- Represent as a linear acceptor, $H$. 
Our Approach: FST-based Reconstructor

- Compose with the Reduction FST, S.
- S is a single-state FST that takes reduced graphemes as input and produces original graphemes as output.
- For example,
Our Approach: FST-based Reconstructor

- Further compose with the **Edit Distance FST**, $E$.
- $E$ is an FST that takes a grapheme sequence as input. It produces as output all grapheme sequences that satisfy the constraint that every word in the output is within an edit distance of $d$ from each word in the input. The allowable edits are substitutions, insertions and deletions.
- Each edit incurs an additive cost $\lambda$.
- $d$ and $\lambda$ are hyperparameters.
Our Approach: FST-based Reconstructor

- Further compose with the **Dictionary FST**, $L$.
- We fix a vocabulary; in this case, the set of all ASR training set words.
- $L$ simply maps a sequence of graphemes to a sequence of words (each word is internally represented as an index in the aforementioned vocabulary).
- Out-of-vocabulary words are mapped to a special `<unk>` word.
Our Approach: FST-based Reconstructor

- Further compose with the **Language Model FST**, $G$.
- $G$ is an n-gram language model trained on ASR training set transcriptions.
- $H \circ S \circ E \circ L$ contains all possible reconstructions. Composing this with $G$ rescores the reconstructions, giving higher scores to meaningful sentences.
- These operations are efficient owing to highly-optimized FST libraries.
Our Approach: FST-based Reconstructor

- Finally, obtain output $O$, the best reconstructed sequence, by running a shortest path FST algorithm on the composed FST $H \circ S \circ E \circ L \circ G$.
- These operations are efficient owing to highly-optimized FST libraries.
Experiments

- **2 Indian languages**: Gujarati, Telugu
- **ASR architecture**: biLSTM (without and with RNNLM)
- **2 Training Durations**: Full and 10-hr
- Gujarati 10-hr experiments on the advanced Conformer ASR architecture
Experimental Setup: BiLSTM (without RNNLM)

**biLSTM Architecture for Speech Recognition**

We use the [ESPNet](#) toolkit to train hybrid CTC-attention biLSTMs

Major hyperparameters:
- 4 encoder layers: 512 units for Guj, 768 units for Tel
- 1 decoder layer: 300 units for Guj, 450 units for Tel
- 0.8 CTC, 0.2 Attention

Reference:
K. Audhkhasi, G. Saon, Z. Tüske, B. Kingsbury and M. Picheny, “Forget a Bit to Learn Better: Soft Forgetting for CTC-Based Automatic Speech Recognition,” in Interspeech, 2019.
Experimental Setup: FSTs

- All FSTs were implemented using the OpenFST toolkit.
- The LM FST, $G$, is a 4-gram LM with Kneser-Ney discounting for order 4. It is implemented using SRILM.
- Best tuned values: $d=3$, $\lambda=5$. 
Results: Pre-Reconstruction ASR Experiments

| Duration | Reduction  | r-WER (Guj) | r-WER (Tel) |
|----------|------------|-------------|-------------|
|          |            | Dev | Test | Dev | Test |
| Full     | identity   | 41.5| 43.2 | 44.1| 46.8 |
|          | $\rho_1$   | 36.5| 39.6 | 39.3| 42.8 |
|          | $\rho_1$-rand | 41.3| 42.3 | 44.2| 47.9 |
| 10 hr    | identity   | 60.2| 68.6 | 64.1| 71.4 |
|          | $\rho_1$   | 53.9| 63.6 | 56.9| 66.5 |
|          | $\rho_1$-rand | 63.2| 71.8 | 60.8| 69.4 |

Reduced Word Error Rate (r-WER) (WERs computed between ASR hypothesis and reduced ground truth text)

*Identity*: Baseline with no reduction

*$\rho_1$*: Our reduction

*$\rho_1$-rand*: Randomized reduction
## Results: Pre-Reconstruction ASR Experiments

| Duration | Reduction   | r-WER (Guj) | r-WER (Tel) |
|----------|-------------|-------------|-------------|
|          | Dev        | Test        | Dev        | Test        |
| Full     | identity   | 41.5        | 43.2        | 44.1        | 46.8        |
|          | $\rho_1$   | 36.5        | 39.6        | 39.3        | 42.8        |
|          | $\rho_1$-rand | 41.3        | 42.3        | 44.2        | 47.9        |
| 10 hr    | identity   | 60.2        | 68.6        | 64.1        | 71.4        |
|          | $\rho_1$   | 53.9        | 63.6        | 56.9        | 66.5        |
|          | $\rho_1$-rand | 63.2        | 71.8        | 60.8        | 69.4        |

- Lower r-WERs for $\rho_1$ show that reduction **simplifies** the ASR task.
- $\rho_1$ vs $\rho_1$-rand shows that a **principled reduction** is important.
Results: Post-reconstruction

| d | λ | Reduction | WER (Guj) | WER (Tel) |
|---|---|-----------|-----------|-----------|
|   |   | Baseline  | Dev 41.5  | Test 43.2 |
|   |   |           | Dev 44.1  | Test 46.8 |
| 0 | 5 | identity  | 41.8      | 43.4      |
|   |   | ρ₁        | 40.4      | 41.9      |
| 3 | 5 | identity  | 37.9      | 37.8      |
|   |   | ρ₁        | 37.8      | 36.5      |

(a) Full training duration.

| d | λ | Reduction | WER (Guj) | WER (Tel) |
|---|---|-----------|-----------|-----------|
|   |   | Baseline  | Dev 60.2  | Test 68.6 |
|   |   |           | Dev 64.1  | Test 71.4 |
| 0 | 5 | identity  | 60.3      | 68.6      |
|   |   | ρ₁        | 56.2      | 64.9      |
| 3 | 5 | identity  | 56.8      | 64.9      |
|   |   | ρ₁        | 53.2      | 61.2      |

(b) 10-hr training duration.

Word Error Rate (WER) for different values of d and λ

ρ₁ is our approach.
Results: FST Reconstruction

| d | λ | Reduction | WER (Guj) | WER (Tel) |
|---|---|-----------|-----------|-----------|
|   |   | Baseline  | Dev 41.5  | Test 43.2 |
| 0 | 5 | identity  | 41.8     | 43.4      |
|   |   | ρ₁        | 40.4     | 41.9      |
| 3 | 5 | identity  | 37.9     | 37.8      |
|   |   | ρ₁        | 37.8     | 36.5      |

(a) Full training duration.

| d | λ | Reduction | WER (Guj) | WER (Tel) |
|---|---|-----------|-----------|-----------|
|   |   | Baseline  | Dev 60.2  | Test 68.6 |
| 0 | 5 | identity  | 60.3     | 68.6      |
|   |   | ρ₁        | 56.2     | 64.9      |
| 3 | 5 | identity  | 56.8     | 64.9      |
|   |   | ρ₁        | 53.2     | 61.2      |

(b) 10-hr training duration.

- For $d=0$ (exact reconstruction), reduction outperforms identity and baseline.
- Increasing $d$ improves all WERs as expected; reduction still outperforms the other two.
- Improvements are more pronounced in the low-resource 10-hr setting.
Experimental Setup: biLSTM (with RNNLM)

- 2 RNNLM layers with 1500 units
- Trained on transcriptions of full speech data
Results: With RNNLM

| Duration | Reduction | WER (Guj) | WER (Tel) |
|----------|-----------|-----------|-----------|
|          | Baseline  | Dev 37.4  | Test 34.0 | Dev 37.9 | Test 40.0 |
| Full     | identity  | 36.2      | 31.8      | 37.7     | 39.2      |
|          | $\rho_1$  | 37.1      | 32.2      | **36.5** | **38.1**  |
| 10-hr    | Baseline  | 56.2      | 63.2      | 56.9     | 63.8      |
|          | identity  | 55.5      | 62.3      | 56.2     | 62.5      |
|          | $\rho_1$  | **52.0**  | **58.2**  | **51.2** | **59.1**  |

Word Error Rate (WER) using reconstructor with $d=3, \lambda=5$ on ASR with RNNLM rescoring
Results: With RNNLM

| Duration | Reduction | WER (Guj) | WER (Tel) |
|----------|-----------|-----------|-----------|
|          | Baseline  | Dev: 37.4 | Test: 34.0 | Dev: 37.9 | Test: 40.0 |
| Full     | identity  | 36.2      | 31.8      | 37.7      | 39.2      |
|          | $\rho_1$  | 37.1      | 32.2      | 36.5      | 38.1      |
| 10-hr    | Baseline  | 56.2      | 63.2      | 56.9      | 63.8      |
|          | identity  | 55.5      | 62.3      | 56.2      | 62.5      |
|          | $\rho_1$  | 52.0      | 58.2      | 51.2      | 59.1      |

- Baseline with RNNLM is **better** than baseline without RNNLM
- Reduction significantly **outperforms** identity in the 10-hr setting, doesn’t do as well in the Full setting for Guj
Experimental Setup: Conformer

**Conformer** Architecture for Speech Recognition

We use the [ESPNet](https://espnet.github.io/) toolkit to train hybrid CTC-attention Conformers.

Major hyperparameters:
- 2 encoder layers: 350 units, 4 att heads
- 1 decoder layer: 350 units, 4 att heads
- 0.3 CTC, 0.7 Attention

Reference:
A. Gulati, J. Qin, C-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu and R. Pang, “Conformer: Convolution-augmented Transformer for Speech Recognition” in Interspeech, 2020.
Results: Conformer on Guj 10-hr

| d | λ Reduction | WER (Guj) |
|---|-------------|-----------|
|   |             | Baseline  | Dev   | Test  |
| 0 | 10 identity |           | 57.7  | 61.1  |
|   | ρ₁          |           | 57.9  | 60.4  |
| 3 | 10 identity |           | 57.1  | 60.5  |
|   | ρ₁          |           | 57.6  | 59.9  |

Similar trends as for other experiments
Discussion

- **Choice of reduction**: We show in the paper that our reduction is superior to randomized/less compressive reductions.
- **Reduction function corrects ASR errors**: 16.29% (for Gujarati) and 16.92% (for Telugu) of identity substitutions errors corrected by the reduction.
- **Test-set perplexities**: Reduction function decreases LM perplexity. Larger drop for Telugu corresponds to larger improvements observed for Telugu.

| Reduction | Test ppl (Guj) | Test ppl (Tel) |
|-----------|----------------|----------------|
| identity  | 115.05         | 768.66         |
| $\rho_1$  | 108.13         | 706.32         |
Discussion

● Examples:

R: సపానా తీసి పాతాల చదువు చేసి యుద్ధం చేసి
   (సపా: na: te: prətə:p ja:dəve: ji:tɨ che:)
I: సప మాట తీసి పాతాల చదువు చేసి
   (సపా: maː te: prətə:p ja:dəv li:dhi che:)
ρ₁: సపానా తీసి పాతాల చదువు చేసి
   (సపా: na: te: prətə:p ja:dəve: ji:tɨ che:)

R: కెదర ముడ పూర్తి మహారాజ
   (iːtaku veɻi ba:luɖi mruti)
I: అనొక్క ముడ కొండ మహారాజ
   (inka: veɻi bo:lo mruti)
ρ₁: కెదర ముడ పూర్తి మహారాజ
   (iːtaku veɻi na ba:luɖi mruti)
Future Work

- Automatically learning a data-driven reduction mapping.
- Training more powerful sequence-to-sequence reconstruction modules
- Combine the two modules into one using a bottleneck layer and multitask learning.
- Instead of the ASR 1-best hypothesis, use the ASR decoding lattice.
Conclusion

- We propose a simple reduce-and-reconstruct technique and demonstrate its utility for two Indian languages.
- We show that as the available training data decreases, our approach yields greater benefits, making it well-suited for low-resource languages.
Short Presentation Slides
Reduce and Reconstruct:
ASR for Low-Resource Phonetic Languages

Anuj Diwan, Preethi Jyothi
Department of Computer Science and Engineering,
Indian Institute of Technology Bombay, India
Reduce and Reconstruct (RnR)

- Technique to boost end-to-end (E2E) ASR performance on low-resource languages:
  a. Train an E2E ASR system with a linguistically-motivated reduced output alphabet (*reduce*)
  b. Train a standalone FST-based reconstructor that recovers sequences in the original alphabet (*reconstruct*)
- Experiments on Gujarati and Telugu.
- With access to only 10 hrs of speech data, we obtain relative WER reductions of up to 7% compared to baseline systems.
Our Approach

1. **Devise a reduced vocabulary** that merges acoustically confusable and linguistically discriminative graphemes.
Our Approach

2. Given labelled speech data, **transform transcriptions** using the reduction.
3. **Train** an **ASR system** that maps the original speech to the reduced transcriptions.

Sound wave saying ભાષા
Our Approach

4. **Train a reconstructor** to reconstruct the original grapheme sequence.
Our Approach: FST-based Reconstructor

- **Input:** Represent as a linear acceptor, $H$.
- **Compose with a cascade of FSTs:** $S$, $E$, $L$, $G$:
  - Using the reduction, $S$ is able to reconstruct all possible sequences.
  - $L$ and $G$ constrain, rank these sequences using language-model scores.
Our Approach: FST-based Reconstructor

- **Input:** reduced-grapheme hypothesis from ASR system.
- Represent as a linear acceptor, $H$. 

\[
\text{पस}
\]
Our Approach: FST-based Reconstructor

- Compose with the **Reduction FST**, S.
- S is a single-state FST that takes reduced graphemes as input and produces original graphemes as output.
- For example,
Our Approach: FST-based Reconstructor

- Further compose with the **Edit Distance FST**, E.
- E is an FST that takes a grapheme sequence as input. It produces as output all grapheme sequences that satisfy the constraint that every word in the output is within an edit distance of \( d \) from each word in the input. The allowable edits are substitutions, insertions and deletions.
- Each edit incurs an additive cost \( \lambda \).
- \( d \) and \( \lambda \) are hyperparameters.
Our Approach: FST-based Reconstructor

- Further compose with the **Dictionary FST**, L.
- We fix a vocabulary; in this case, the set of all ASR training set words.
- L simply maps a sequence of graphemes to a sequence of words (each word is internally represented as an index in the aforementioned vocabulary).
- Out-of-vocabulary words are mapped to a special `<unk>` word.
Our Approach: FST-based Reconstructor

- Further compose with the **Language Model FST**, $G$.
- $G$ is an n-gram language model trained on ASR training set transcriptions.
- $H \circ S \circ E \circ L$ contains all possible reconstructions. Composing this with $G$ rescores the reconstructions, giving higher scores to meaningful sentences.
- These operations are efficient owing to highly-optimized FST libraries.
Our Approach: FST-based Reconstructor

- Finally, obtain output $O$, the best reconstructed sequence, by running a shortest path FST algorithm on the composed FST $H \circ S \circ E \circ L \circ G$.
- These operations are efficient owing to highly-optimized FST libraries.
Experiments

- **2 Indian languages**: Gujarati, Telugu
- **ASR architecture**: Bi-LSTM (without and with RNNLM)
- **2 Training Durations**: Full and 10-hr
- Gujarati 10-hr experiments on the advanced Conformer ASR architecture
| ASR Architecture | Training-set Duration | Reduction                  | Gujarati Test WER | Telugu Test WER |
|------------------|------------------------|----------------------------|-------------------|-----------------|
|                  |                        | none (baseline)            | 43.2              | 46.8            |
|                  |                        | identity                   | 37.8              | 42.5            |
|                  |                        | our reduction               | 36.5              | 41.2            |
| biLSTM           | Full                   | none (baseline)            | 68.6              | 71.4            |
|                  |                        | identity                   | 64.9              | 66.1            |
|                  |                        | our reduction               | 61.2              | 63.6            |

- Reduction **outperforms** identity and baseline
- Improvements are more pronounced in the **low-resource** 10-hr setting
## Results

| ASR Architecture | Training-set Duration | Reduction            | Gujarati Test WER |
|------------------|------------------------|----------------------|-------------------|
| Conformer        | 10-hr                  | none (baseline)      | 61.1              |
|                  |                        | identity             | 60.4              |
|                  |                        | our reduction        | 59.9              |
## Results

| ASR Architecture | Training-set Duration | Reduction       | Gujarati Test WER | Telugu Test WER |
|------------------|-----------------------|-----------------|-------------------|-----------------|
| biLSTM           | Full                  | none (baseline) | 34.0              | 40.0            |
|                  |                       | identity        | 31.8              | 39.2            |
|                  |                       | our reduction    | 32.2              | 38.1            |
|                  | 10-hr                 | none (baseline) | 63.2              | 63.8            |
|                  |                       | identity        | 62.3              | 62.5            |
|                  |                       | our reduction    | 58.2              | 59.1            |

- Reduction is significantly **better** in the 10-hr setting
- Reduction doesn’t do as well in the Full setting for Gujarati
Analysis

- **Choice of reduction**: We show in the paper that our reduction is superior to randomized/less compressive reductions.

- **Reduction function corrects ASR errors**: 16.29% (for Gujarati) and 16.92% (for Telugu) of identity substitution errors corrected by the reduction.

- **Test-set perplexities**: Reduction function decreases LM perplexity.

| Reduction       | Test ppl (Guj) | Test ppl (Tel) |
|-----------------|----------------|----------------|
| identity        | 115.05         | 768.66         |
| our reduction    | 108.13         | 706.32         |
Discussion

- Examples:

R: రాయి తెష్ప ప్రతాప వాడవెంది చే
   (సేప: నా: తె: ప్రేతా: ప జా: దవె: జి: టి చె:)

I: రాయ మారే తెష్ప ప్రతాప వాడవెంది చే
   (సేప: మా: తె: తె: ప్రేతా: ప జా: దవె లి: డీ చె:)

$p_1$: రాయి తెష్ప ప్రతాప వాడవెంది చే
   (సేప: నా: తె: ప్రేతా: ప జా: దవె: జి: టి చె:)

R: కడింగ ఉండవచే మార్య
   (ఇతకు ఉల్లి బా: లుడి మరుతి)

I: అక్ష కండి మార్య
   (ఇంక: ఉల్లి బా: లడ మరుతి)

$p_1$: కడింగ ఉండవచే మార్య
   (ఇటకు ఉల్లింంచ బా: లుడి మరుతి)
Conclusion and Future Work

- We propose a simple reduce-and-reconstruct (RnR) technique for E2E ASR systems and demonstrate its utility for two phonetic languages.
- As the available training data decreases, RnR yields greater benefits, making it well-suited for low-resource languages.
- Future work includes:
  - Training more powerful sequence-to-sequence reconstruction modules
  - Automatically learning a mapping from the original alphabet to the reduced alphabet