Uniwersytet Wrocławski

TOOPLOOX AI

Team UWr and Tooploox (sharp-perch) solution to Airbus ATC Challenge 2018

Team members

- University of Wroclaw:
  - faculty: Jan Chorowski, Paweł Rychlikowski
  - PhD students: Michał Zapotoczny
  - master students: Igor Gajowiak, Grzegorz Jurdziński, Bartosz Kostka, Bartosz Krzeczkowski, Szymon Malik
- Tooploox:
  - Krzysztof Wróbel, Tomasz Trzciński

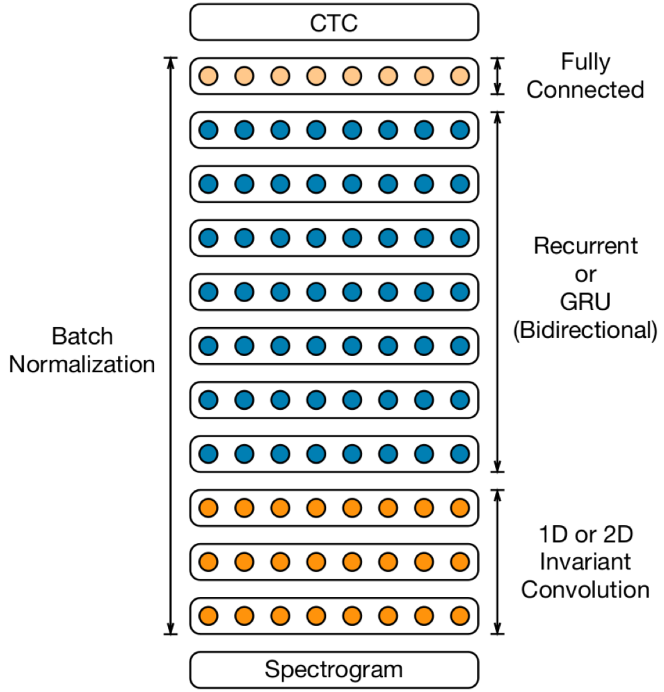
Our model at a glance

- Deep neural net with CTC, Convs and BiLSTMs over Mel Filterbanks
- Predicts phonemes at 33Hz, context handled using biphones
- 4-gram LM trained on training transcripts
- Lattice decoding using Kaldi (CTC mapped to HMM topology)
- Data augmentation (frequency shifts, noising)
- Final model is an ensemble of 6 networks with different hyperparams

Details: Acoustic modelling network

Inspired by Deep Speech 2

- Input: 80 mel-filterbanks at 100Hz with energy, deltas+ddeltas, CMVN
- 2 Conv layers:
  - 7x7 stride 1x2 + 7x7 stride 3x1
  - Batchnorm after each conv
- 5 BiLSTMs
  - 768 units in each direction + batchnorm


CTC
Fully Connected
Recurrent or GRU (Bidirectional)
Batch Normalization
1D or 2D Invariant Convolution
Spectrogram

Details: model training

- Models trained in one step, no dependency on traditional GMM-HMM
- Typical training time: 2 days on a single K40 gpu
- Used mini-batch SGD, learning rate decayed based on dev set performance
- After initial convergence (5000 iterations), wintroduced weight noise
  (at each iterations small Gaussians added to each weight)
- Polyak averaging used for checkpoints
  didn’t improve final results, but gave much faster convergence

Details: Data augmentation

Our final model ensembled networks trained with different data augmentations.
WER decrease 8.5% -> 7.8% on dev set (sampled from train data)

- Audio:
  - Adding random utterance audio as noise, with varied SNR
    we have noticed that some original audio files had background spoken noise
  - Random pitch changes (up to 300% up and down)
  - Speed change (both slowdown and speedup)
- Text:
  - Introduced edit errors:
    - Removing characters
    - Replacing character with random one
    - Swapping neighboring characters
  - This was aimed to prevent network overconfidence (putting all probability mass on just one character per frame, which kill LM integration)

Details: Biphoneme model and lattice decoder

- For CTC training we used Biphoneme targets (previous, current)
- Trained with one giant softmax with $|V|^2$ outputs, no decision tree involved
- Used WFST-based decoding using Kaldi tools:
    - CLG are standard
    - Kaldi’s basic decoder used used a CTC-specific H graph
      (FST that removes duplicates and discards blanks)
    - Kaldi’s lattice decoder required an HMM topology, CTC can be moddeled as a two-states per phone architecture with all blanks tied using the decision tree.
- Final model used lattice decoding, with a slight word insertion bonus and same weighting of acoustic model and language model

Details: Language model and data augmentation

- Standard 4-gram models trained on augmented transcripts form train data.
- Motivation for LM training corpus augmentation:

if one aircraft can do something, then others can do it as well

- LM training corpus generation:
  - Identify “safe” call signs in train data
  - Compute call sign statistics (distribution P_{cs})
  - For every sentence in train data create its variants with call signs replaced by cs drawn from P_{cs}
- This creates new, potentially useful, N-grams not present in learning data
- LM overfits less to a single aircraft doing one thing multiple times in the training corpus

Callsign detection: Overview

- Simple call sign definition:
  - Fixed prefix from https://en.wikipedia.org/wiki/List_of_airline_codes + some contractions found in learning data (royal air instead royal air maroc, …)
  - Followed by some digits and letters (from NATO alphabet)
- (simple) CS finding:
  - Do some shallow parsing identifying phrases with digits and letter which are NOT cs
  - Find maximal CS which does not intersect with phrases from previous step
  - Is parsing useful?
    - **KLM six three tango** five mile final ILS three two right
    - **Germania one eight eight one** three and half miles
    - huh Toulouse **Cargolux triple seven** two thousand five hundred feet climbing level seven zero

Callsign detection: shallow parsing

- Simple recursion free grammar describing some phrases in text:

```
A  -> alpha|bravo|charlie|...|zulu
D  -> one|two|...|nine|niner

RUNWAY_IN_USE -> runway in use D D left|right
RUNWAY_IN_USE -> runway D D in use

QFE -> qfe D D D
QFE -> quebec fox|foxtrot echo D D D

HOLDING_POINT -> holding point A D D left|right
HOLDING_POINT -> holding point A D D D left|right
HOLDING_POINT -> holding point A D
```

- Approx. 250 handcrafted rules
- Constructs that matched were usually restricted from being parts of callsigns

Callsign detection: Details

- Problem: some CS does not fulfill simple definition:
  - two six delta lima turn right heading zero eight zero (Air Hop omitted)
  - Civil aircraft call signs (only letters and digits)
- Our solution:
  after shallow parsing try to detect promising digits and letters sequences at the beginning or at the end of utterance
- *Future work:*
  Add more machine learning to call sign detection
- *(perhaps more reliable data will be needed)*

Takeaways: What did work for us

- CTC: fast training from scratch
- Biphone output modelling
- Polyak checkpoint averaging: speedy model convergence
- Data augmentation
  - both for Acoustic and Language models
  - key differentiator between models in the ensemble
- Lattice decoding and ensembling of lattices
- Future work with LM: separate ATIS from rest of the corpora, apply text data augmentation with other ‘categories’ (wind information, temperature, speed)
- Using bigger context can be very useful (for instance repeated call signs)

Takeaways: What didn’t work

- Uniform train/dev/test data split
  - had same conversation in all splits
    overfitting due to speaker and phrase overlap
  - better to split by date or location
- Data denoising
  - need to train on denoised data too, otherwise results prone to deteriorate
  - better to train the net to implicitly denoise by data augmentation
- Two-pass decoding with LM adaptation hurt
- Training on data transformed with scoring rules hurt
  (e.g. we replaced all *nine*s with *niner*s)
- However: no observed benefits from training on disambiguated transcripts
  (some transcripts used *niner*s for clearly audible *nine*s)

Conclusions

- This was a fun competition to participate - thanks!
- There are lots of new research directions
    - Speaker adaptation: so far our model is speaker independent
    - Use the context and assume that speech comes from conversation (useful e.g. for callsign detection)

Acknowledgements

We thank:

- National Science Center (Poland) for Grant Sonata 2014/15/D/ST6/04402
- PLGrid project for computational resources on Prometheus cluster