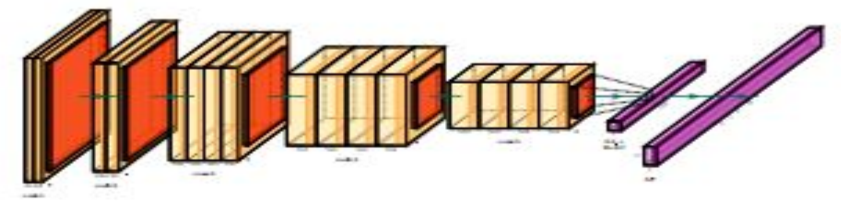
Number and Keyword Spotting

Speech YOLO Algorithm - The Model

This Algorithm uses a Computer Vision Inspired CNN (Convoluted Neural Network) which is able to localise and identify specific keywords

<u>Model</u>: The model consists of 16 convolution layers and 2 fully connected layers, in addition to this Batch Normalisation and ReLU is applied after each layer.



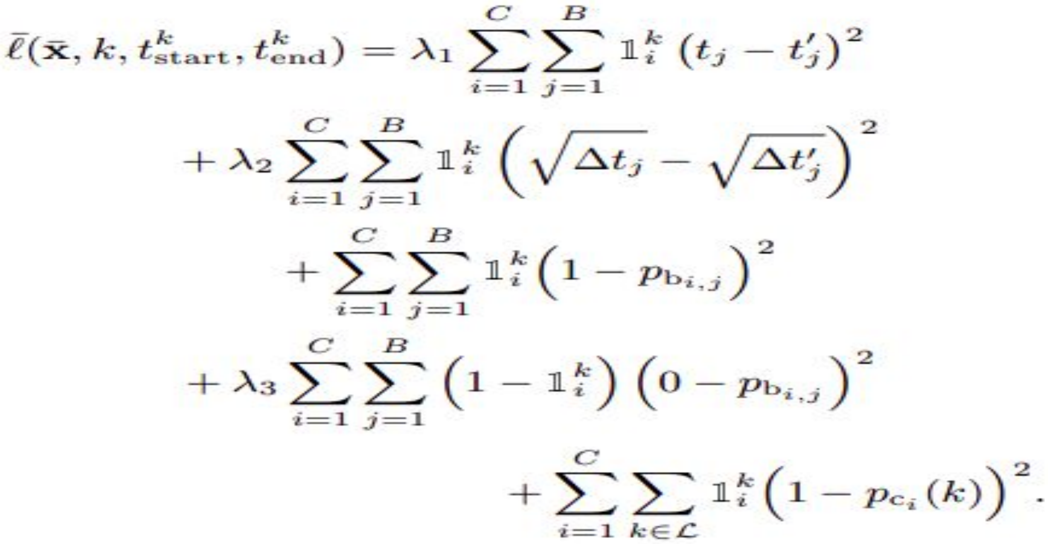
There are multiple different models that can be employed but the best one is the VGG19 C_B_K compatible version.

Feature Extraction and Loss Function

Audio Features are extracted by reading the sound file resulting in a 2D feature array of size (frames x channels)

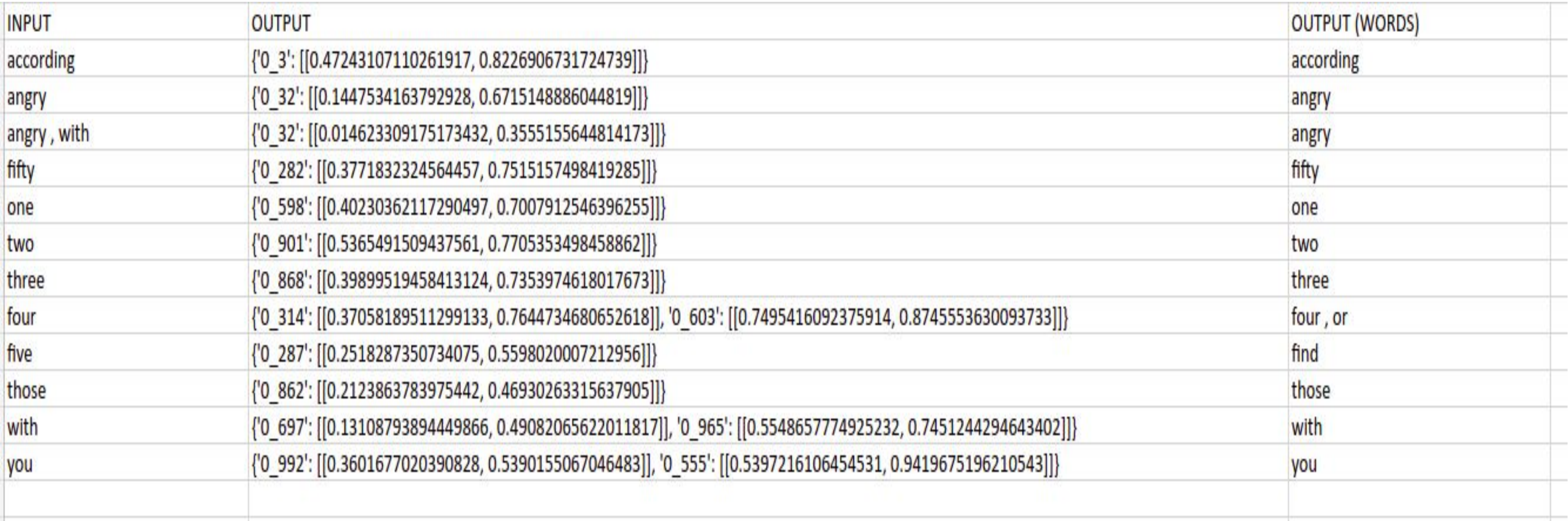
Pre-Processing: The data is then preprocessed by applying Short Term Fourier Transform (STFT), followed by normalization.

Loss Function: The Model's loss function is calculated as the sum of various different aspects of the model as follows-



$$
\begin{aligned}
\bar{\ell}(\bar{\mathbf{x}}, k, t^k_{\mathrm{start}}, t^k_{\mathrm{end}}) = {} & \lambda_1 \sum_{i=1}^{C} \sum_{j=1}^{B} \mathbb{1}_i^k \left(t_j - t'_j\right)^2 \\
& + \lambda_2 \sum_{i=1}^{C} \sum_{j=1}^{B} \mathbb{1}_i^k \left(\sqrt{\Delta t_j} - \sqrt{\Delta t'_j}\right)^2 \\
& + \sum_{i=1}^{C} \sum_{j=1}^{B} \mathbb{1}_i^k \left(1 - p_{\mathrm{b}_{i,j}}\right)^2 \\
& + \lambda_3 \sum_{i=1}^{C} \sum_{j=1}^{B} \left(1 - \mathbb{1}_i^k\right) \left(0 - p_{\mathrm{b}_{i,j}}\right)^2 \\
& + \sum_{i=1}^{C} \sum_{k \in \mathcal{L}} \mathbb{1}_i^k \left(1 - p_{\mathrm{c}_i}(k)\right)^2 .
\end{aligned}
$$

A few experiments



INPUT	OUTPUT	OUTPUT (WORDS)
according	{'0_3': [[0.47243107110261917, 0.8226906731724739]]}	according
angry	{'0_32': [[0.1447534163792928, 0.6715148886044819]]}	angry
angry , with	{'0_32': [[0.014623309175173432, 0.3555155644814173]]}	angry
fifty	{'0_282': [[0.3771832324564457, 0.7515157498419285]]}	fifty
one	{'0_598': [[0.40230362117290497, 0.7007912546396255]]}	one
two	{'0_901': [[0.5365491509437561, 0.7705353498458862]]}	two
three	{'0_868': [[0.39899519458413124, 0.7353974618017673]]}	three
four	{'0_314': [[0.37058189511299133, 0.7644734680652618]], '0_603': [[0.7495416092375914, 0.8745553630093733]]}	four , or
five	{'0_287': [[0.2518287350734075, 0.5598020007212956]]}	find
those	{'0_862': [[0.2123863783975442, 0.46930263315637905]]}	those
with	{'0_697': [[0.13108793894449866, 0.49082065622011817]], '0_965': [[0.5548657774925232, 0.7451244294643402]]}	with
you	{'0_992': [[0.3601677020390828, 0.5390155067046483]], '0_555': [[0.5397216106454531, 0.9419675196210543]]}	you

Fuzzy Matching Algorithm - on transcripts

This process matches the transcripts with the keywords using 2 parameters, **bigram match** and **soundex** score.

Soundex:

- A phonetic indexing algorithm which matches words with similar pronunciation.
- Can also match words across different languages having similar pronunciation.
- Returns 1 or 2 if words match depending on language and -1 if they do not.

Bigram Match:

- Returns a score based on the number of common bigrams between the 2 strings.

Final Score:

- The final match score is calculated based on the 2 factors and they keyword having the highest score above a threshold gets detected

Examples

English:

Please transfer two thousand four hundred and fifty two rupees to my account (Input)

please transfer {two thousand four hundred and fifty two} rupees to my account (Output)

Bengali:
অনুগ্রহ করে আমার অ্যাকাউন্টে দুই হাজার চারশ বায়ান্ন টাকা ট্রান্সফার করুন (Input)
অনুগ্রহ করে আমার অ্যাকাউন্টে {দুই হাজার চারশ বায়ান্ন} টাকা ট্রান্সফার করুন (Output)

References

Soundex: https://github.com/libindic/soundex

Fuzzy Matching: https://github.com/libindic/inexactsearch

Speech YOLO : https://github.com/MLSpeech/speech_yolo

Speech YOLO paper : https://arxiv.org/pdf/1904.07704.pdf