# DATS 6303 - Deep Learning Final Presentation

# LipReader

**PRESENTED BY:**
Hala, Sayam & Henry

# Agenda

| | |
|---|---|
| 03 | Introduction |
| 04 | Dataset |
| 05 | Data Pre-processing |
| 06 | Experimentation |
| 07 | Final Model |
| 08 | Product Demo |
| 09 | Conclusions |

# Introduction and Literature Review

**We were inspired by LipNet the first end-to-end sentence-level lipreading model.** Using that as a foundation, we built our own models that take short video clips, to perform **end-to-end lipreading at sentence level**, not just isolated words.

Combines **CNNs**, **bidirectional GRUs**, and **CTC loss** to map raw video frames → text.

**Why Lip Reading?**

There are several reasons to invest in lip reading technology:

- **Accessibility:** Helps deaf or hard-of-hearing individuals understand speech and enables assistive applications.
- **Security and Surveillance:** Can decode speech silently, useful in situations where audio recording is unavailable or impractical.



## LipNet: End-to-End Sentence-level Lipreading

**Yannis M. Assael**[1,†], **Brendan Shillingford**[1,†], **Shimon Whiteson**[1] **& Nando de Freitas**[1,2,3]
Department of Computer Science, University of Oxford, Oxford, UK [1]
Google DeepMind, London, UK [2]
CIFAR, Canada [3]
{yannis.assael,brendan.shillingford,
shimon.whiteson,nando.de.freitas}@cs.ox.ac.uk

### Abstract

Lipreading is the task of decoding text from the movement of a speaker's mouth. Traditional approaches separated the problem into two stages: designing or learning visual features, and prediction. More recent deep lipreading approaches are end-to-end trainable (Wand et al., 2016; Chung & Zisserman, 2016a). However, existing work on models trained end-to-end perform only word classification, rather than sentence-level sequence prediction. Studies have shown that human lipreading performance increases for longer words (Easton & Basala, 1982), indicating the importance of features capturing temporal context in an ambiguous communication channel. Motivated by this observation, we present LipNet, a model that maps a variable-length sequence of video frames to text, making use of spatiotemporal convolutions, a recurrent network, and the connectionist temporal classification loss, trained entirely end-to-end. To the best of our knowledge, LipNet is the first end-to-end sentence-level lipreading model that simultaneously learns spatiotemporal visual features and a sequence model. On the GRID corpus, LipNet achieves 95.2% accuracy in sentence-level, overlapped speaker split task, outperforming experienced human lipreaders and the previous 86.4% word-level state-of-the-art accuracy (Gergen et al., 2016).

### 1 Introduction

Lipreading plays a crucial role in human communication and speech understanding, as highlighted by the McGurk effect (McGurk & MacDonald, 1976), where one phoneme's audio dubbed on top of a video of someone speaking a different phoneme results in a third phoneme being perceived.

Lipreading is a notoriously difficult task for humans, specially in the absence of context[1]. Most lipreading actuations, besides the lips and sometimes tongue and teeth, are latent and difficult to disambiguate without context (Fisher, 1968; Woodward & Barber, 1960). For example, Fisher (1968) gives 5 categories of visual phonemes (called *visemes*), out of a list of 23 initial consonant phonemes, that are commonly confused by people when viewing a speaker's mouth. Many of these were asymmetrically confused, and observations were similar for final consonant phonemes.

Consequently, human lipreading performance is poor. Hearing-impaired people achieve an accuracy of only $17 \pm 12\%$ even for a limited subset of 30 monosyllabic words and $21 \pm 11\%$ for 30 compound words (Easton & Basala, 1982). An important goal, therefore, is to automate lipreading. Machine lipreaders have enormous practical potential, with applications in improved hearing aids, silent dictation in public spaces, security, speech recognition in noisy environments, biometric identification, and silent-movie processing.

Machine lipreading is difficult because it requires extracting spatiotemporal features from the video (since both position and motion are important). Recent deep learning approaches attempt to extract those features end-to-end. Most existing work, however, performs only word classification, not sentence-level sequence prediction.

[†]These authors contributed equally to this work.
[1]LipNet video: https://youtube.com/playlist?list=PLXkuFIFnXUAPIrXKgtIpctv2NuSo7xw3k

# Understanding the Data

**Dataset Overview:** Used GRID audiovisual sentence corpus developed in 2006 by the Centre for Speech Technology Research, University of Edinburgh.

**Purpose:** Designed for audio-visual speech recognition research, to study lipreading and multimodal speech processing.

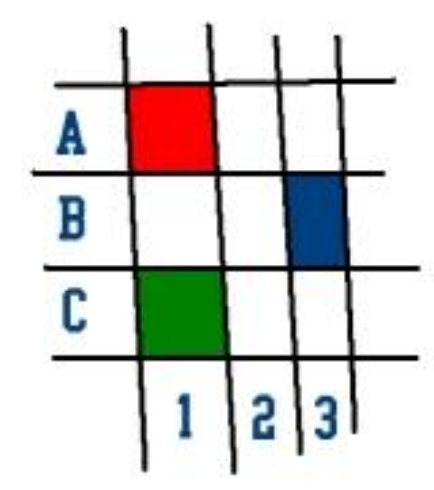
**Design Rationale:**

- **Controlled vocabulary & syntax:** Each sentence follows a fixed six-word structure to reduce linguistic variability.
- **Multiple speakers & repetitions:** Ensures variability in lip movements while allowing robust training/testing.
- **Aligned video and audio:** Facilitates precise mapping from visual features (lip movements) to words.
- **Standardized recording conditions:** Fixed camera, lighting, and background to minimize confounding factors.

# Data Structure

- **Speakers:** 34 speakers (male and female), each with 1,000 video recordings.
- **Sentence Format:** Each sentence contains 6 words: **command, color, preposition, letter, digit, adverb** (e.g., “Place red at B 9 now”).
- **Video Files:** (.mpg) format, usually 3-4 seconds long, frontal face view.
- **Alignment Files:** (.align) files per video containing word-level timestamps for precise audio-to-visual mapping.

```
0 23750 sil
23750 29500 bin
29500 34000 blue
34000 35500 at
35500 41000 f
41000 47250 two
47250 53000 now
53000 74500 sil
```



# Data Preprocessing

**Goal:** Prepare aligned lip-region frames with transcripts for model training.

**Data Pipeline Overview:**

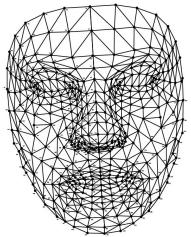
1. **Data Pairing:** Matched .mpg videos with .align files.
2. **Feature Extraction:** Used automated face- and mouth-region detection methods to locate the speaking area, crop the relevant region of interest, resize frames to a consistent resolution, and normalize pixel values to produce stable, standardized inputs.
3. **Storage:** Saved processed frames as **.npy arrays, transcripts as .txt,** and maintain a manifest (.pkl) containing (frames_path, transcript_path) tuples for dataset loading.

# Feature Extraction

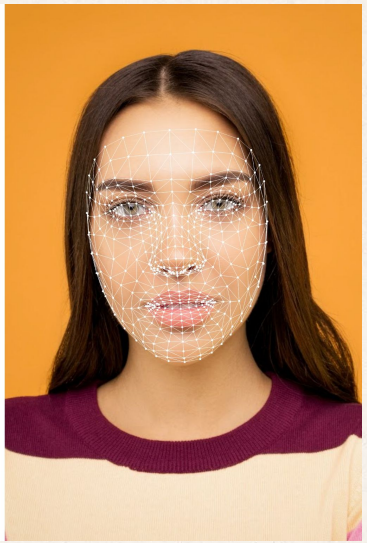
**Why?**
Cropped lip regions focus the model on relevant mouth movements, reduce background noise, and support consistent input learning.

**Approaches we used:**

- **Automatic Cropping:** Default dimensions provided from the GRID Corpus dataset (46, 140) to identify the mouth.
- **MediaPipe Face Mesh (Google - Deep Learning):** Provides 468 dense 3D facial landmarks with high temporal stability, enabling lip-region localization and consistent preprocessing for our lip-reading networks. *Its fine-grained detail is highly effective for capturing subtle mouth movements, though it is more computationally intensive and can be sensitive to occlusions.*



468 points

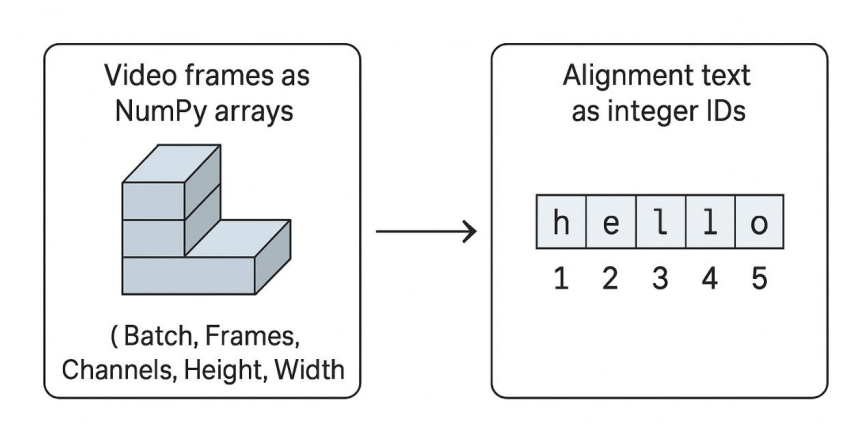


# Input and Target Processing for Model Training

After cropping and **resizing** the mouth frames, we **normalized** pixel intensities, **standardized** video length and converted it to **grayscale** to standardize the input distribution during model training.


Next, we converted the video frames from Numpy arrays into PyTorch tensors of shape **(Batch, Frames, Channels, Height, Width)** to feed into our model.

The corresponding alignment text was encoded into integer IDs using a **character-to-index mapping**, enabling the use of **CTC loss** for sequence prediction. This mapping was saved for consistent tokenization during evaluation and testing, ensuring the model could correctly interpret predictions back into characters.



# Evaluation Metrics

**Character Error Rate (CER):** Measures the total character-level edits (insertions, deletions, substitutions) needed to match predictions to references, normalized by total reference length.

**Word Error Rate (WER):** Measures accuracy at the word level, capturing sequence-level errors.

**Computation:**

1. Decode model logits using greedy decoding.
2. Compare predictions to ground-truth transcripts.
3. Compute CER/WER using edit distance between predicted and reference sequences.

$$\text{CER} = \frac{\text{Number of character edits (insertions + deletions + substitutions)}}{\text{Total number of characters in reference}}$$

$$\text{WER} = \frac{\text{Number of word edits (insertions + deletions + substitutions)}}{\text{Total number of words in reference}}$$

**Loss Metrics:** Training, Validation, and Test loss (CTC Loss) to monitor model convergence and stability.

# Loss Function

### Why We Chose CTC (Connectionist Temporal Classification Loss)

- Handles **variable-length sequences**: videos have many frames, text sequences are shorter.
- Eliminates the need for **frame-to-character alignment**, simplifying preprocessing.
- Widely used in **speech recognition and lip-reading** tasks, proven to work with deep sequence models.
- Works with **CNN-LSTM architectures**, allowing end-to-end training.

$$P(y|x) = \sum_{\pi \in \text{Align}(y)} P(\pi|x)$$

### How We Implemented It

- Based on PyTorch's nn.CTCLoss.
- Input to the loss function includes:
  - logits: model predictions ([time, batch, num_classes])
  - targets: ground-truth text encoded as integer IDs
  - input_lengths: number of frames per video
  - target_lengths: number of characters per text sequence
- Returns a **scalar loss**.

$$\text{Loss} = -\log P(y|x)$$

# Experimentation

## Model #1

**Dataset Subset:** 1000 Videos. Speaker 22 (s22) female

**Model Overview:**

- LipNet-style Sequential model in Keras.
- 3D convolutional layers extract mouth movement features.
- Dense output layer with softmax predicts characters + CTC blank token.

**Architecture:**

- Conv3D #1: 128 filters → ReLU → MaxPool3D
- Conv3D #2: 256 filters → ReLU → MaxPool3D
- Conv3D #3: 75 filters → ReLU → MaxPool3D
- Dense Output: Softmax

**Training & Hyperparameters:**

- **Loss:** CTC Loss
- Trained until convergence with early stopping.

**Results & Interpretation:**

- Validation loss ≈ 18.78.
- Model captures mouth movement patterns relevant for phoneme prediction.
- Architecture and preprocessing support generalization on unseen clips.
- High loss suggests need for more data, hyperparameter tuning, or extended training for improved accuracy.

# Experimentation

## Model #2

**Dataset Subset:** 5000 Videos.

## Architecture

1. **3D CNN Layers**
   - Extract **spatial features** from each frame (edges, mouth shapes, movements).
   - 3 blocks with increasing filters: 32 → 64 → 128.
   - Each block: Conv → BatchNorm → ReLU → MaxPool → Dropout.
2. **Bidirectional LSTM**
   - Processes temporal sequence of frame features.
   - Captures **how mouth shapes change over time**.
   - Bidirectional: learns context from past and future frames.
3. **Attention Module**
   - Computes a **weighted sum of LSTM outputs**.
   - Focuses on frames most informative for predicting current character.
   - Improves alignment between video frames and predicted text.
4. **Fully Connected Layer ( LayerNorm & Dropout)**
   - Maps features to **logits over all characters and CTC blank token “no character”.**
   - Prepares predictions for **CTC loss**, which handles unaligned sequences.

# Experimentation

**Model & Loss**

- 3D Convolutional → LSTM → Attention → Fully Connected layer.
- Outputs **logits per character per frame**.
- Used **CTC loss** to handle variable-length sequences without explicit alignment.

**Training Hyperparameters**

- **Batch Size:** 2, 8
- **Learning Rate:** 3e^-4
- **Optimizer:** AdamW
- **Epochs:** 100

**Decoding & Evaluation**

- Used **greedy decoding** on logits, remove repeated characters and blank tokens.
- **Metrics: Character Error Rate (CER)** and **Word Error Rate (WER)** for performance evaluation.

**Validation**

- **Validation**: Compute logits → decode predictions → calculate CER/WER.
- **Early stopping**: monitors validation loss, saves best model.

# Results

- **CER (~0.55):** Shows the model is partially learning character-level patterns, but many predictions are still incorrect.
- **WER (~0.95):** Indicates the model fails to produce coherent words, with most word predictions being wrong.
- **Model Behavior:** Training and validation losses are close, suggesting no overfitting, but the model underfits, improvements may require more data, hyperparameter tuning, or better decoding strategies.

**Training**

- **Train Loss:** 1.5697
- **Val Loss:** 1.5968
- **CER:** 0.5655
- **WER:** 0.9354

**Test**

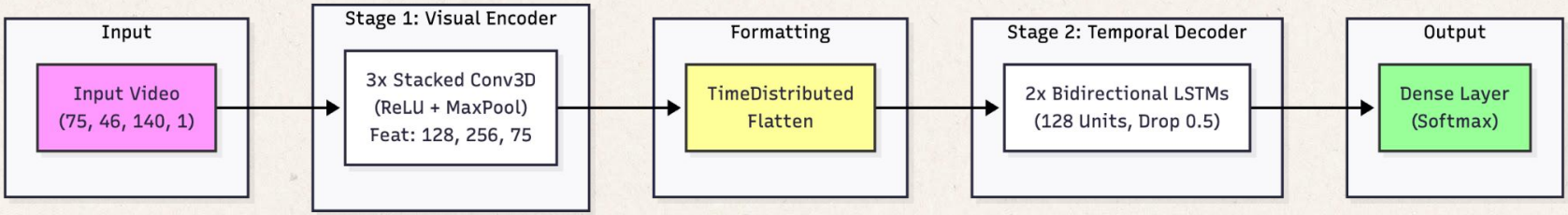
- **CER:** 0.5486
- **WER:** 0.9522

# Final Model

**Model Architecture:** CRNN with three stacked 3D convolutional blocks (128, 256, 75 filters) using ReLU activation and (1,2,2) max-pooling to extract spatial features while preserving temporal depth. Temporal dependencies are captured by two Bidirectional LSTM layers (128 units each) with orthogonal initialization and 50% dropout, followed by a dense softmax layer for character predictions.

**Input & Feature Extraction:** Processes video inputs of shape (75,46,140,1) with 3×3×3 convolution kernels; 3D convolutions downsample spatial dimensions while keeping temporal information intact.

**Regularization & Training:** Employs dropout (0.5) and batch normalization to prevent overfitting and stabilize training; designed to handle variable-length temporal sequences effectively.



# Results

- **CER (~0.49 vs ~0.54 previously):** Slight increase in character-level errors, showing the model is partially learning characters but still makes many mistakes.
- **WER (~0.87 vs ~0.95 previously):** Noticeable improvement in word-level predictions, though a significant portion of words remain incorrect.
- **Model Behavior:** Training and validation losses remain close (Train: 3.4 vs Val: 1.6), indicating no overfitting; underfitting persists, suggesting the need for more data, hyperparameter tuning, or improved decoding strategies.

**Training**

- **Train Loss:** 3.4
- **Val Loss:** 1.6
- **CER:** 0.4938
- **WER:** 0.8745

**Test**

- **CER:** 0.5063
- **WER:** 0.9045

# Conclusions & Next Steps

## Summary

- We were able to demonstrate the feasibility of character-level video-to-text lipreading using spatiotemporal neural networks inspired by *LipNet (Assael et al., 2016)*!
- Preprocessing pipeline (face detection, lip localization, grayscale conversion, normalization, sequence standardization) produced consistent, stable inputs.
- All models trained stably and learned basic mouth-movement patterns, but **underfitting persisted**, leading to **high CER/WER** and limited word-level coherence.

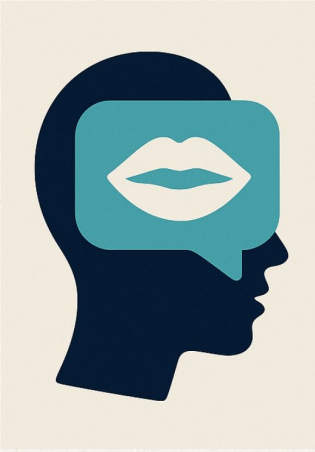
## Key Learnings

- Dataset limitations were the primary bottleneck, models learned general mouth-motion patterns but could not generalize to full word-level structure. This was due to computational limitations.
- Preprocessing quality is critical, accurate face detection, lip localization, and consistent frame normalization significantly improved training stability.
- Attention helps highlight informative frames, but improvements are limited when training data lacks diversity.

# Conclusions & Next Steps

## Next Steps

- Expand dataset size and diversity (more speakers, lighting conditions, angles, sentence structures).
- Explore stronger architectures (Transformers, improved CNN-LSTM hybrids).
- Improve decoding with **beam search** or integration of a **language model**.
- Apply further data augmentation, hyperparameter tuning, and enhanced preprocessing.



# Product Demo

# Thank you!