# Large speech and language models in prosody research

Sofoklis Kakouros

11 April 2024

# Agenda

A. Investigating the interplay between word predictability, prominence, and emotion recognition in Large Language Models (LLMs) as well as the prosodic information that LLMs encode:
   1. *prosodic information in BERT,*
   2. *word surprisal and emotion recognition in speech,*
   3. *word surprisal, prominence, and speech synthesis.*

B. Exploring the use of Large Speech Models for prosodic analysis tasks:
   1. *dialect identification.*

C. Development of new pooling methods from Large Speech Models for prosodic tasks:
   1. *correlation pooling,*
   2. *attentive correlation pooling.*

# A. Word predictability, prominence, and emotion recognition with LLMs

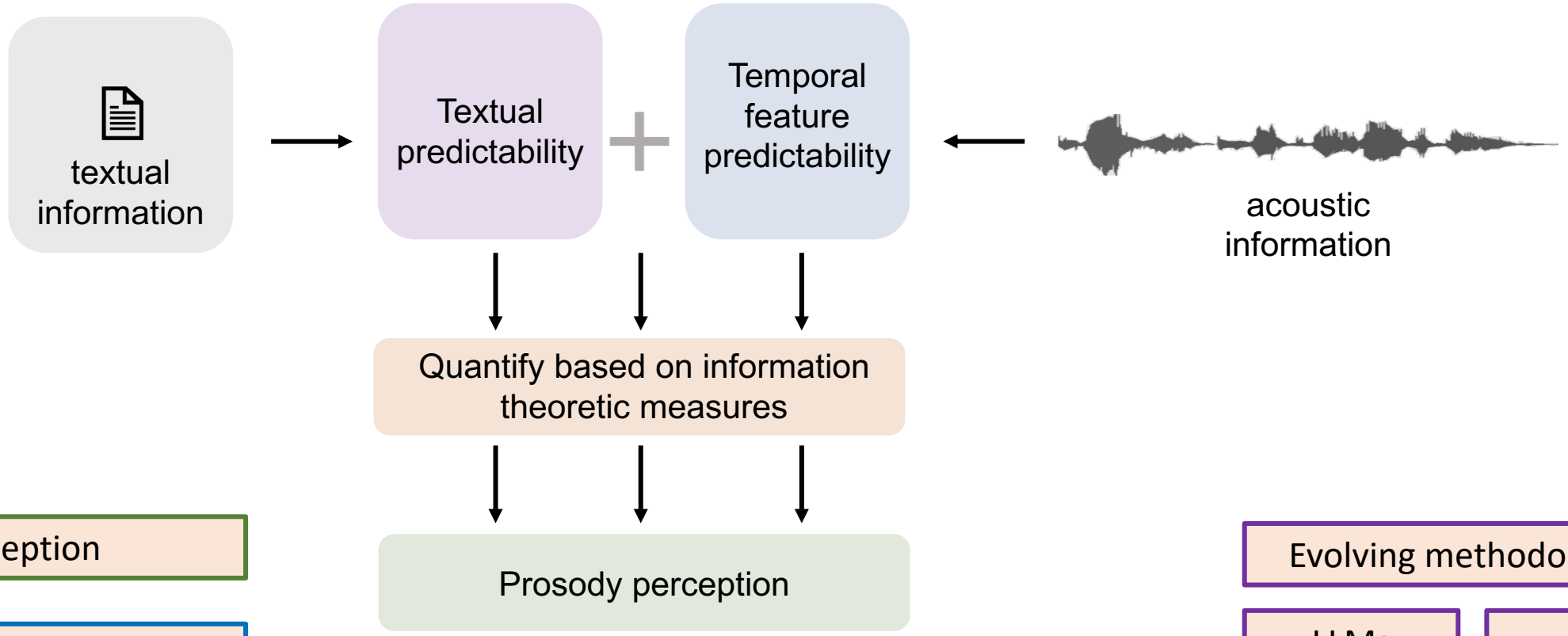
# Information theory as a means of understanding prosody

# Guiding research questions

- ***How acoustic and linguistic variation is statistically organized into perceived prosody?***

- ***How can we harness the identified relationships to advance the capabilities of speech technology applications?***

# Overview of the research approach



# Prosody in large speech and language models

- Representations from large language and speech models have become nearly ubiquitous in state-of-the-art speech and language applications.
- These models have achieved state-of-the-art results also in prosody tasks.
- As these models do not define predetermined linguistic targets during training, we need to address the following questions:
  - ***Is prosody linguistically encoded in LLMs?***
  - ***What kind of prosodic characteristics do large speech models encode?***

Kakouros, S. and O'Mahony, J. (2023). What does BERT learn about prosody? In R. Skarnitzl, & J. Volín (Eds.), *Proceedings of the 20th International Congress of Phonetic Sciences (ICPhS-2023)* (pp. 1454-1458). GUARANT International spol. s r.o.., Prague, Czechia.

# Prosody in BERT?

- Language models (LMs) are nearly ubiquitous in natural language processing applications.
- LM design does not define predetermined linguistic targets during training.
- Several studies have explored the linguistic information that models capture providing some insights on their representational capacity.
- However, there have not been investigations exploring whether prosody is part of the structural information of the language that models learn.
- ***Is prosody linguistically encoded in BERT?***

# Layer weights

- We obtain the contribution of BERT layers to the prediction task by introducing learnable scalar weights attached to each transformer layer of the model.
- We take representations from all transformer layers in the model and collapse them to one via a weighted average.
- There is one weight for each layer and all weights are trained jointly with the classification network.

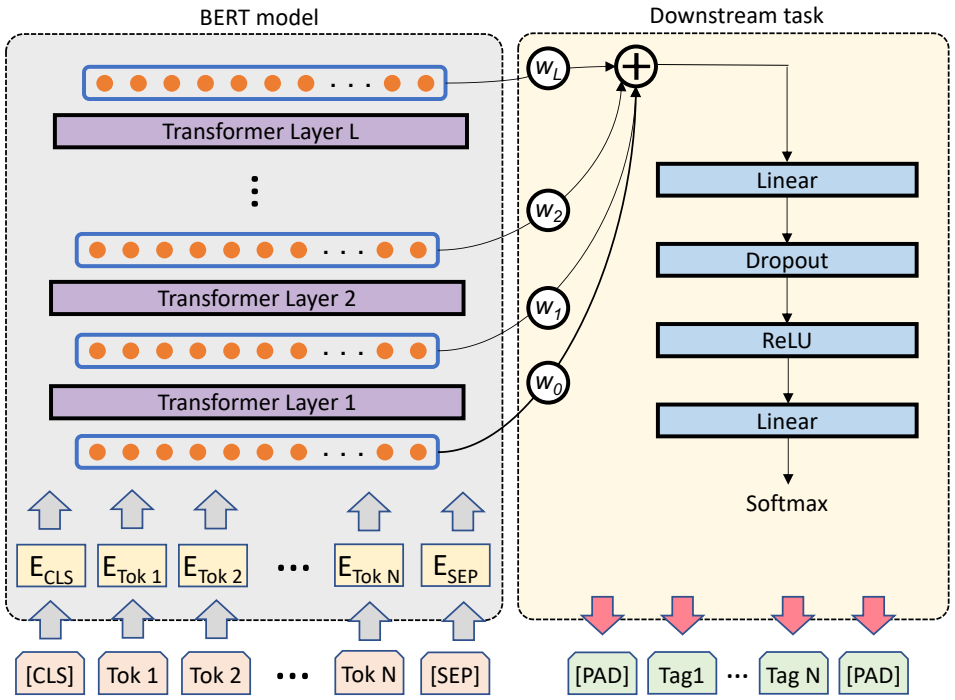



Fig. Architectural overview

*bert-base-uncased; 12 layers, embedding size of 768*

# Layer weights

- To probe the contribution of individual BERT layers, we extract the embeddings from each layer separately.
- We then train a classification head consisting of a single dense layer followed by the Softmax function.
- For each embedding layer, we then obtain the classification accuracy for the task.

# Data

- In the experiments we use three datasets: two consisting of read speech and one of spontaneous dialogue speech:
    1. **BURNC**: The Boston University Radio News Corpus is a corpus of professionally read news data in American English annotated with ToBI labels.
    2. **NXT Switchboard**: NXT is a dataset that includes dialogues from the Switchboard corpus annotated with ToBI labels.
    3. **LibriTTS**: The LibriTTS is a read speech corpus (subset of the LibriSpeech corpus). Prosody labels are automatically generated and are taken from the Helsinki Prosody Corpus (HPC).

# Experimental setup

- Prominence and POS prediction.
- POS tags extracted with spaCy.
- 80-15-5 split for train, validation, and test.
- Training run for 20 epochs with a batch size of 4.
- Each experiment repeated five times.
- Results are averaged.
- We test both frozen and fine-tuned models.

# Results

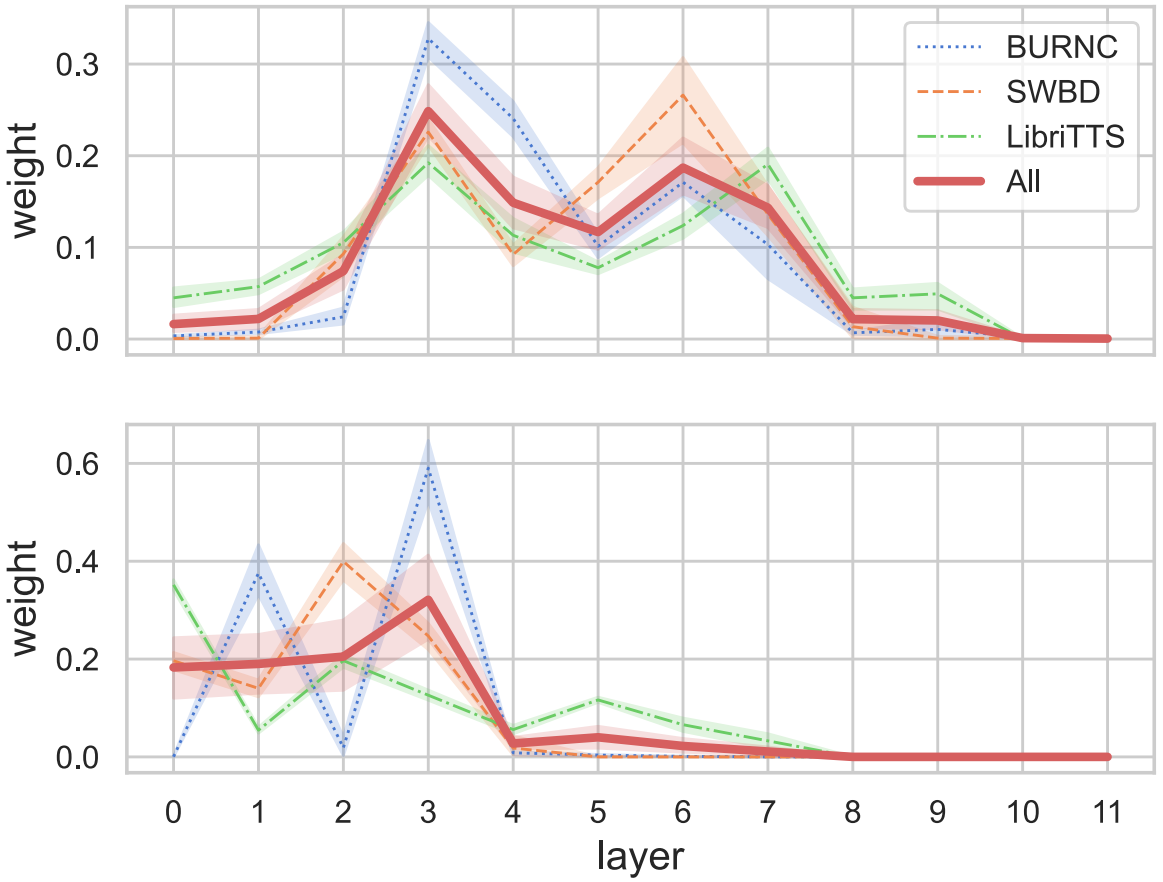



Fig. BERT layer weights for prominence (top) and POS (bottom) prediction.

# Results

**Table 1:** Layer-wise accuracy for the test set runs for prominence and POS prediction with frozen BERT. Numbers in bold denote the two top results in each task.

| *Layer* | **BURNC** | | **SWBD** | | **LibriTTS** | |
|---|---|---|---|---|---|---|
| | *Prom* | *POS* | *Prom* | *POS* | *Prom* | *POS* |
| 0 | 81.51 | 93.84 | 75.32 | **94.62** | 80.18 | 89.50 |
| 1 | 83.52 | **93.90** | **76.14** | **94.67** | **80.32** | **90.08** |
| 2 | 85.20 | **94.33** | **76.01** | 94.02 | 80.29 | **89.53** |
| 3 | 85.20 | 93.84 | 75.47 | 93.50 | **81.08** | 89.00 |
| 4 | **85.26** | 93.47 | 75.25 | 92.52 | 80.28 | 88.40 |
| 5 | **85.26** | 93.17 | 75.82 | 91.87 | 80.00 | 87.29 |
| 6 | 84.13 | 92.98 | 75.71 | 90.95 | 79.71 | 86.12 |
| 7 | 84.46 | 91.76 | 75.82 | 90.11 | 79.40 | 84.86 |
| 8 | 83.79 | 91.03 | 75.45 | 88.37 | 79.05 | 83.48 |
| 9 | 82.92 | 89.20 | 74.71 | 87.20 | 78.63 | 82.36 |
| 10 | 82.59 | 88.47 | 74.29 | 86.64 | 78.32 | 81.14 |
| 11 | 83.26 | 87.19 | 73.08 | 82.09 | 77.27 | 76.67 |

**Table 2:** Accuracy for the test set runs for prominence and POS prediction with frozen and fine-tuned (ft) BERT.

| *Prominence* | **BURNC** | **SWBD** | **LibriTTS** |
|---|---|---|---|
| frozen | 87.14 | 78.10 | 82.67 |
| ft | 85.53 | 75.95 | 80.32 |
| *POS* | | | |
| frozen | 95.97 | 97.94 | 98.49 |
| ft | 97.56 | 98.54 | 98.96 |

# Results

- Layer weights vary with respect to the two tasks.
- The three datasets show differences in their overall performance.
- However, the overall weight allocation for each task is consistent.
- POS information is encoded in the early BERT layers.
- Most POS information comes from layers 0-4 which have been shown to encode surface linguistic features.
- Weights for prominence are primarily focused within layers 2-8 with a peak appearing at layer 3
- Fine-tuning leads to better performance.

# Conclusions

- BERT captures information about prosodic prominence through a widespread allocation of weights across its layers reaching high performance.
- The weight allocations suggest that BERT relies on a variety of linguistic information including surface features such as POS but also syntactic and semantic information.

Kakouros, S. (submitted). Enhancing Speech Emotion Recognition through Word Informativeness. Submitted to the *Annual Conference of the International Speech Communication Association (Interspeech-2023)*

# Emotions and word informativeness

- In emotion recognition from speech, a key challenge lies in identifying speech signal segments that carry the most relevant acoustic variations for discerning specific emotions.
- Traditional approaches compute functionals for features such as energy and F0 over entire sentences or longer speech portions, potentially missing essential fine-grained variation.
- ***Can we identify semantically important segments using word informativeness derived from LLMs?***

# Background

- The challenges in SER are multifaceted but can generally be split into three main areas:
  1. development of representations that can accurately and robustly encapsulate the acoustic variation indicative of various emotions,
  2. modelling the temporal dimension of emotions, which can manifest over short or extended speech sequences,
  3. inherent ambiguity of emotional expressions leading to high disagreement in annotating emotions from speech.
- In this work we focus on [2].

# Data

- The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) was used in this study.
- RAVDESS is a multimodal database consisting of recordings from 24 professional actors (12 female) in North American accent.

Livingstone, S. R., & Russo, F. A. (2018). The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. *PloS one*, *13*(5), e0196391.

# Methods – feature extraction

- Acoustic representations were extracted using:
  - Wav2vec 2.0:
    - representations are extracted using *Wav2vec 2.0 base*. The features were extracted from the pre-trained Wav2vec 2.0 by taking the representation from the last transformer layer of the model. To obtain an utterance-level description, the feature-level representations are pooled by taking the mean and standard deviation (std) of the features over each speech segment.
  - eGeMAPS:
    - includes 88 different features and feature functionals including means and standard deviations of F0, loudness, spectral tilt, and MFCCs. The 88 eGeMAPS features and functionals are computed for each selected speech segment, with one single feature vector representing the entire segment.

# Methods – word metrics

- Word probabilities are computed using predictions from GPT-2 small (gpt2; 124M parameters),
- For unigram computation publicly available word counts were used derived from the *Google Web Trillion Word Corpus*.
- Word rank is computed by taking the position in a sorted list of all possible words, ordered by their predicted probabilities (from highest to lowest) by the LLM.

# Methods – word metrics

- Word surprisal can be seen as an information-theoretic measure of the amount of new information conveyed by a word.
- Why use word rank? A word's rank can change based on the context and the probabilities of other words, even if its own probability remains the same.
- For each word in a sentence, the following are computed:
  - Unigram surprisal
  - LLM surprisal (word surprisal values are extracted by taking the aggregate token surprisal)
  - Normalized word rank

# Methods – segmentation

- Speech segments of variable lengths were selected by sorting the words within each sentence according to their surprisal value or rank (e.g., from high to low surprisal).
- For each sentence, words were ordered by their surprisal or rank, and the first *n* words were isolated from the speech signal for subsequent feature extraction.
- Two distinct approaches were considered:
  - *top-n*, wherein the speech segments corresponding to the top *n* ordered words are concatenated and then subjected to feature extraction, and
  - *independent-n*, wherein only the speech segment of the word at position *n*, based on its surprisal or rank, is extracted from the sentence for feature extraction.

# Methods – model training

- Features were used to train a feed-forward Deep Neural Network (DNN) classifier (four hidden layers; [256, 128, 64, 32]) with emotions as the target classes.
- The data were selected to leave two unseen speakers out in the test set while the remaining speakers were used for training and validation.
- The splits were designed in order to have an equal proportion of female and male speakers in both the train and test sets.
- Each split of the dataset had a different set of speakers in the test set.

# Results

Table 1: *Mean accuracy and F1 scores for the test set runs for unigram surprisal (SR), LLM surprisal, and rank-based word selection for Wav2vec 2.0. top denotes the top n words together while ind the word independently at position n. Numbers in bold denote the best results in each metric and task.*

| *top* | **unigram SR** | | **LLM SR** | | **rank** | |
|---|---|---|---|---|---|---|
| | *ACC* | *F1* | *ACC* | *F1* | *ACC* | *F1* |
| 1 | 57.81 | 56.15 | 60.97 | 59.28 | 57.31 | 55.43 |
| 2 | 58.06 | 56.68 | 60.64 | 59.23 | 59.06 | 57.67 |
| 3 | 61.65 | 60.75 | 60.81 | 59.46 | 62.39 | 60.94 |
| 4 | 60.41 | 59.39 | **63.23** | **61.90** | 61.49 | 60.78 |
| 5 | **62.48** | **61.20** | 61.15 | 59.71 | 62.07 | 60.56 |
| 6 | 61.82 | 60.65 | 61.73 | 60.40 | **62.49** | **61.25** |
| *ind* | *ACC* | *F1* | *ACC* | *F1* | *ACC* | *F1* |
| 1 | **57.81** | **56.15** | **60.97** | **59.29** | 57.31 | 55.44 |
| 2 | 57.22 | 55.72 | 55.74 | 53.76 | 58.13 | 56.31 |
| 3 | 56.80 | 55.02 | 59.48 | 57.69 | 59.22 | 57.18 |
| 4 | 13.28 | 3.35 | 13.28 | 3.35 | 13.28 | 3.35 |
| 5 | 57.32 | 55.42 | 13.28 | 3.35 | 13.28 | 3.35 |
| 6 | 13.28 | 3.35 | 13.28 | 3.35 | **59.81** | **58.06** |

Table 2: *Mean accuracy and F1 scores for the test set runs for unigram surprisal (SR), LLM surprisal, and rank-based word selection for eGeMAPS. top denotes the top n words together while ind the word independently at position n. Numbers in bold denote the best results in each metric and task.*

| *top* | **unigram SR** | | **LLM SR** | | **rank** | |
|---|---|---|---|---|---|---|
| | *ACC* | *F1* | *ACC* | *F1* | *ACC* | *F1* |
| 1 | 43.35 | 41.48 | 44.25 | 42.20 | 41.34 | 39.78 |
| 2 | 46.26 | 44.79 | 46.28 | 44.42 | 45.10 | 43.32 |
| 3 | 52.21 | 50.67 | 51.95 | 50.55 | 50.62 | 49.98 |
| 4 | **54.23** | **53.31** | 50.95 | 50.19 | 50.38 | **50.09** |
| 5 | 53.13 | 51.64 | 49.46 | 48.23 | 47.61 | 46.64 |
| 6 | 52.80 | 51.41 | **51.95** | **51.08** | **50.70** | 49.28 |
| *ind* | *ACC* | *F1* | *ACC* | *F1* | *ACC* | *F1* |
| 1 | 43.35 | 41.48 | 44.25 | 42.20 | 41.34 | 39.78 |
| 2 | **43.69** | **42.54** | 13.28 | 3.35 | 40.10 | 38.19 |
| 3 | 41.68 | 39.76 | **44.85** | **43.53** | 13.28 | 3.35 |
| 4 | 13.28 | 3.35 | 13.28 | 3.35 | 13.28 | 3.35 |
| 5 | 13.28 | 3.35 | 13.28 | 3.35 | 13.28 | 3.35 |
| 6 | 13.28 | 3.35 | 13.28 | 3.35 | **52.29** | **50.56** |

# Results

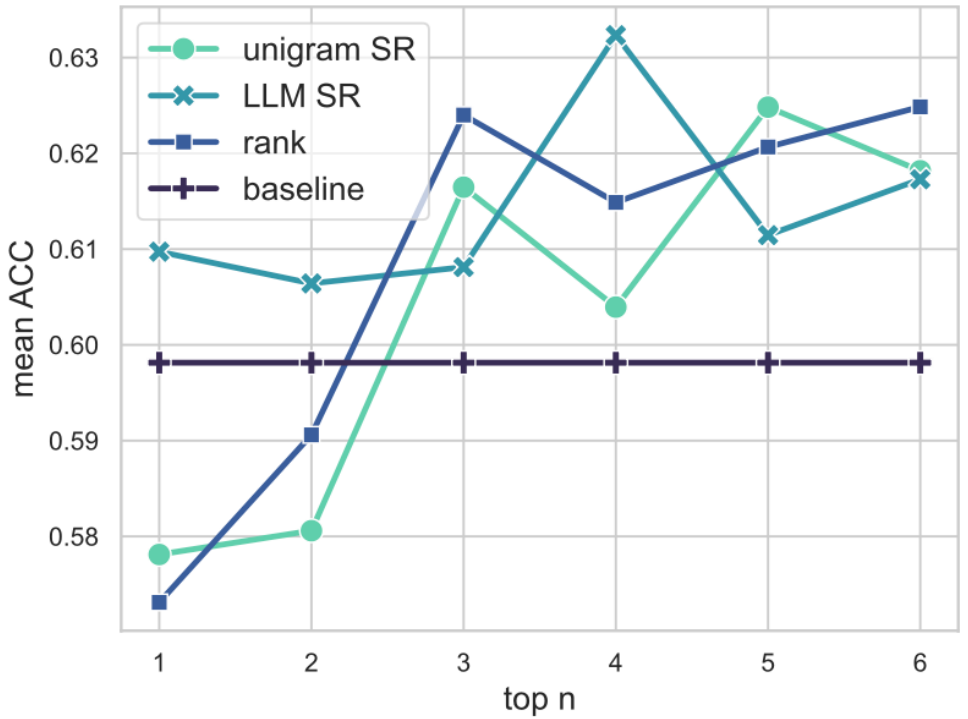



Figure 1: *Mean accuracies over test runs for the top n experiment using Wav2vec 2.0 representations.*

# Conclusions

- Using word informativeness as extracted from an LLM such as GPT-2 can lead to improved performance.
- Using the few most informative words in a sentence to extract acoustic features is sufficient, leading to performance close to that of processing the entire utterance.
- Using few words (utilizing a word selection criterion based on surprisal) rather than the entire utterance was more beneficial for model training leading to improved performance.
- ***Word informativeness seems to provide a useful tool in the aid of temporal segment selection from speech for SER.***

*Kakouros, S., Šimko, J., Vainio, M., and Suni, A. (2023). Investigating the Utility of Surprisal from Large Language Models for Speech Synthesis Prosody. In Proceedings of the 12th ISCA Speech Synthesis Workshop (SSW-2023), Grenoble, France, pp. 127–133. 10.21437/SSW.2023-20*

# Word-level surprisal from pre-trained LLMs

- Does word surprisal, extracted from LLMs, correlate with prominence, a signal-based measure of the salience of a word in a given discourse?

- If this is true, can we use word surprisal as a feature to aid speech synthesis prosody?

# Introduction

- While text-to-speech (TTS) has become indistinguishable from human speech for short utterances, it still struggles with contextual prosody.
  - ❑ Selecting appropriate style, loudness or emotion.
  - ❑ Applying proper word accentuation patterns.

- Why?
  - ❑TTS models are largely trained from text-audio pairs of isolated sentences.
  - ❑This limits the quantity and quality of the linguistic data the models are exposed to.

➢How can we address this issue?

# Links between prosody and surprisal

- We investigate word surprisal, a measure of the predictability of a word in a given context, as a feature to aid speech synthesis prosody

- Surprisal has been shown in the literature to be connected to the impression of highlighting, and in prosodic terms, to the phenomenon of prosodic prominence (Kakouros & Räsänen, 2016)

- In general, (semantic) predictability has been shown to correlate negatively with prominence in speech (cf. “smooth signal redundancy” of Aylett & Turk, 2004)

# Language modeling

- State-of-the-art large language models (LLMs):
  - ❑Can process and generate coherent responses from information that spans much more than a single sentence.
    - ❑GPT-2 can manage contexts up to 1024 tokens.
    - ❑GPT-3 up to 2048 tokens.
    - ❑GPT-3.5 can process up to 4096 tokens.
  - ❑Trained on vast amounts of diverse textual data far surpassing the textual data a TTS has been exposed to during training.
    - ❑GPT-2 has been trained on the WebText dataset that consists of 40Gb of text.

➢Can we use word-level surprisal calculated using LLMs to improve context-dependent realization of prosodic prominence patters?

# Givenness

- Given words, i.e., the words and concepts already previously introduced in a narrative, can be expected to be realized with less prominence than the novel ones (Watson et al., 2006; Sridhar et al., 2008; Féry & Ishihara, 2010).

- Surprisal-based models operating on a large context can be expected to account for this type of givenness, attributing higher probability (lower surprisal) to the words that had occurred previously in the text, particularly for content words.

# Givenness

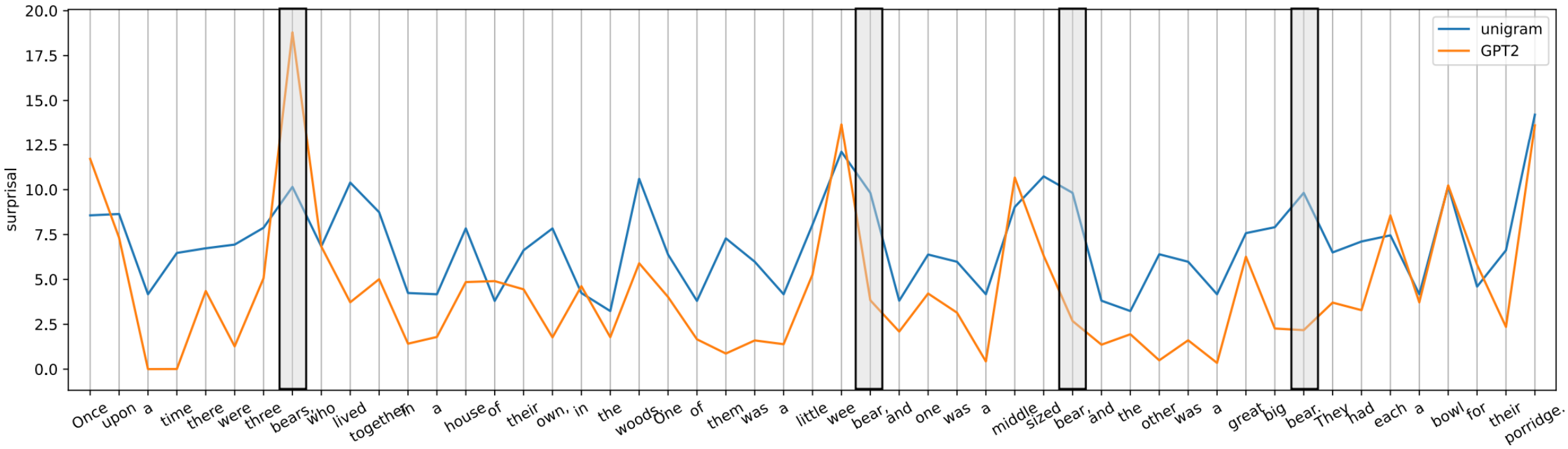



Figure 1: *Context-dependence of GPT surprisal values compared with unigrams.*

# Method

- Data:
  - ❑LJ Speech dataset
- Word surprisal was estimated using predictions from four variants of GPT-2, the GPT-J model, and unigrams:
  - ❑GPT-2 small (gpt2; 124M parameters)
  - ❑GPT-2 medium (gpt2-medium; 355M)
  - ❑GPT-2 large (gpt2-large; 774M)
  - ❑GPT-2 extra large (gpt2-xl; 1.5B)
  - ❑GPT-J (EleutherAI/gpt-j-6b)
  - ❑For **unigram** computation we used publicly available word counts derived from the Google Web Trillion Word Corpus.

# Method

- Context modeling:
  - ❑We construct a context for each target sentence by prepending the target sentence with the text segments that preceded it.
  - ❑For our analysis, we included context of up to 5 previous segments —context sizes from 0 − 5 are denoted in the text as ***sup_0***, ***sup_1***, ***sup_2***, ***sup_3***, ***sup_4***, and ***sup_5*** where 0 refers to a sentence without context.
- Prominence estimation with CWT:
  - ❑To assess the correlation of surprisal values with speech prosody, we utilized word prominence estimates, derived automatically using Wavelet prosody toolkit.

# Method

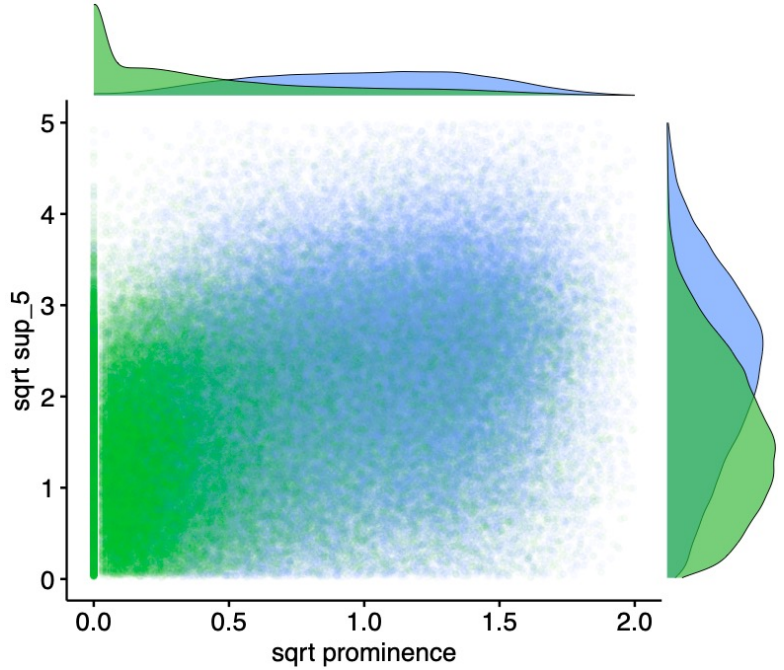
- Speech synthesis:
  - ❑To assess the effect of word surprisal on speech synthesis prosody, we applied a transformer-based FastPitch model architecture augmented with local conditioning.
  - ❑To implement the local conditioning, we repeated the continuous conditioning features (word prominence or surprisal value) for each segmental sound (phone) of the word.
  - ❑We then embedded the features using a linear layer to a dimension of 384 and summed the resulting embeddings with the phone representations of the FastPitch encoder.

# Results

# Signal-based prosodic characteristics and surprisal

EXPECTATION: **The surprisal values will (positively) correlate with prominence estimates and other "prominence yielding features"**

The correlations were calculated for all words in the corpus as well as separately for stop-words and content words, respectively



# Signal-based prosodic characteristics and surprisal

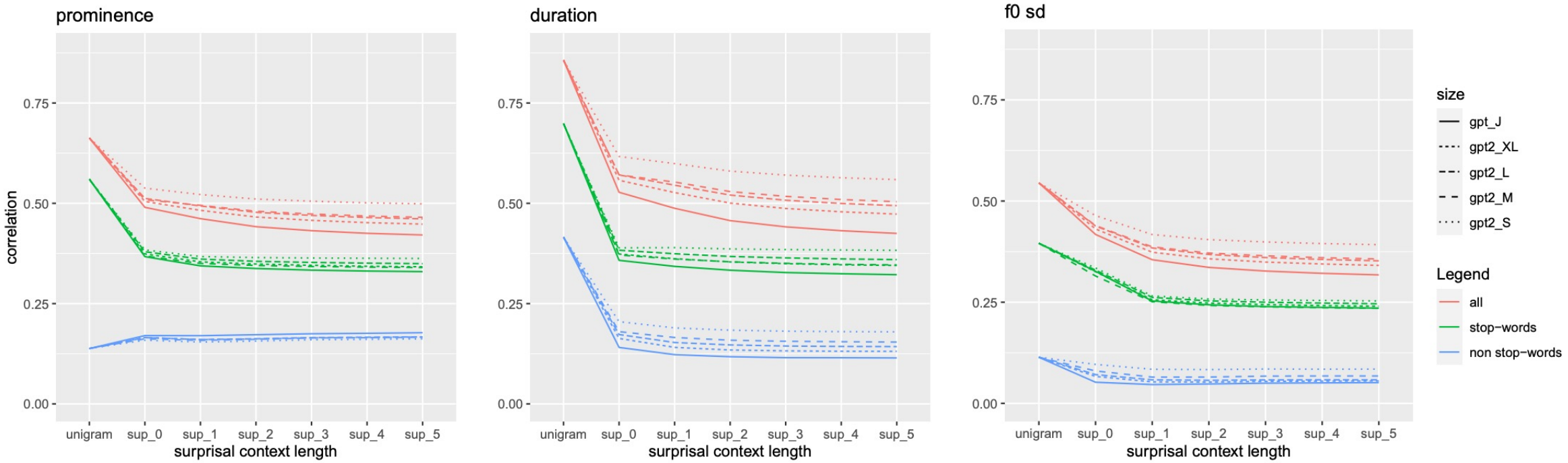



Figure 4: *Spearman's rank correlations between word-level surprisal values and signal-based characteristics for language models of different sizes, for all words in the corpus, for stop-words and for content words. Please note the difference in y-axis scaling between the two rows of plots.*

# Signal-based prosodic characteristics and surprisal

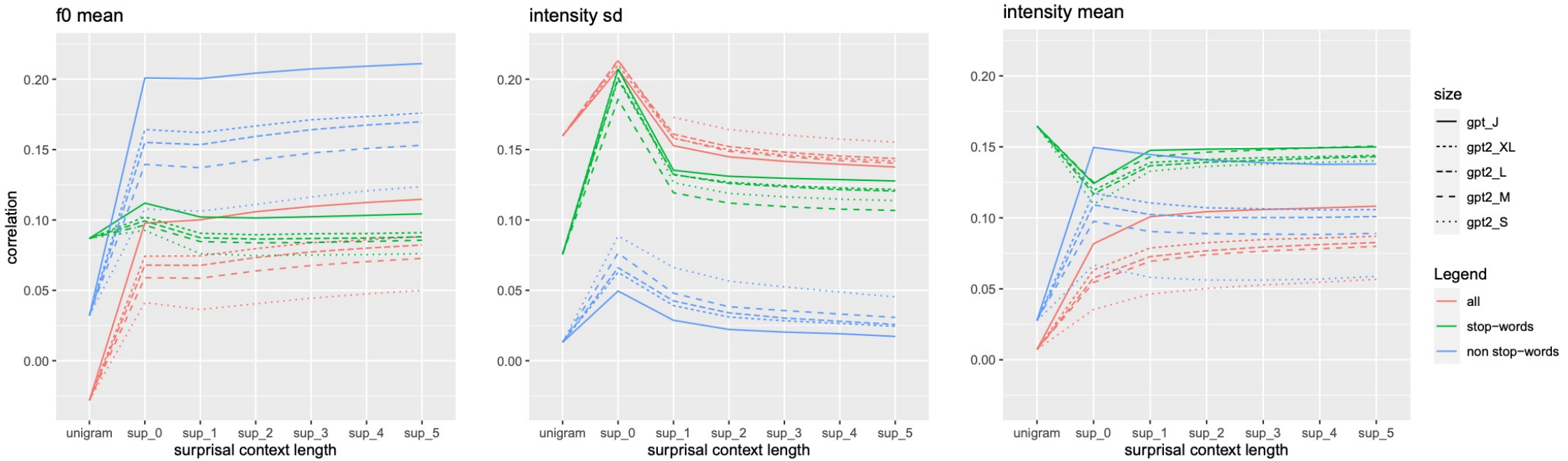



Figure 4: *Spearman's rank correlations between word-level surprisal values and signal-based characteristics for language models of different sizes, for all words in the corpus, for stop-words and for content words. Please note the difference in y-axis scaling between the two rows of plots.*

# Signal-based prosodic characteristics and surprisal

- Correlations are higher for the entire corpus than for the two sub-groups separately (and higher for stop-words than for content words).
- Correlations are generally higher for the smaller models than for the larger ones, and, in most cases, decrease with the size of the context used for the surprisal values.
- The measures of prominence, duration and f0-sd correlate best with unigram-based surprisal.

# Givenness

- To compare the givenness-related prominence patterns with corresponding surprisal values within the analyzed corpus, we assigned each content word in the corpus a distance from its previous occurrence.

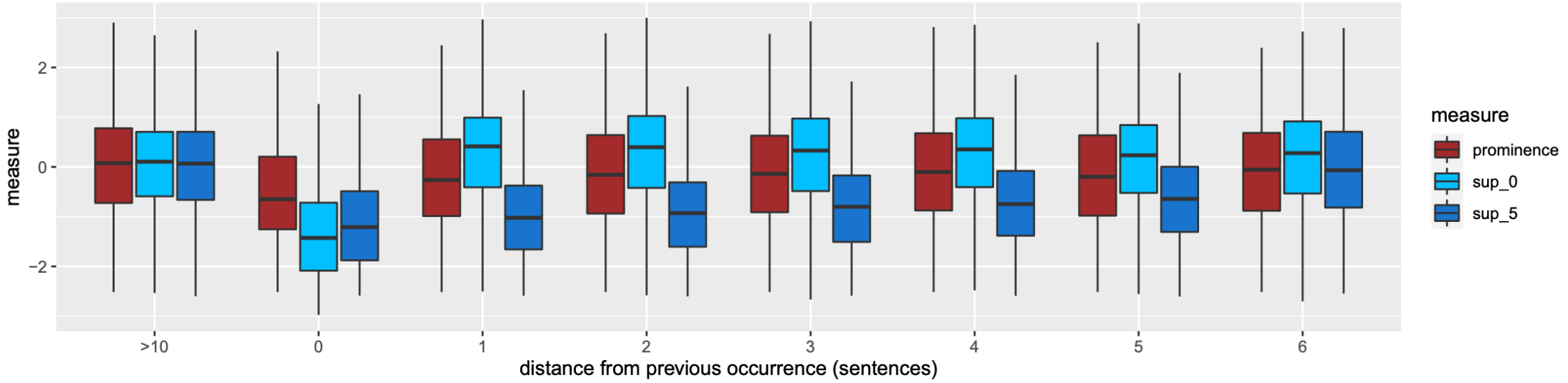



Figure 2: *Dependence of* `prominence`, `sup_0` *and* `sup_5` *values on givenness.*

# Speech synthesis

# Surprisal in speech synthesis

- To assess the effect of word surprisal on speech synthesis prosody, we applied a transformer-based FastPitch model architecture augmented with local conditioning.
- To implement the local conditioning, we repeated the continuous conditioning features (word prominence or surprisal value) for each word.

# Surprisal in speech synthesis

- Two anchor systems were trained:
  - a baseline system with no conditioning and
  - a top line system with CWT prominence conditioning (prom)

- To assess the effect of surprisal, we trained two systems conditioned with word surprisal values:
  - smallest GPT2 with no context (gpt_small_sup0) and
  - GPT_J with context of five previous sentences (gpt_j_sup5).

# Surprisal in speech synthesis

- We synthesized the test material with appropriate conditioning: surprisal values extracted with respective language models with same context sizes as during training.
- For topline model prom, we used prominence labels extracted from the actual speech of the test sentences instead of predicting labels from text.

# Surprisal in speech synthesis

Table 1: *Results of objective speech synthesis evaluation*

| | $f_0$ RMSE | $f_0$ cor | dur RMSE | dur cor |
|---|---|---|---|---|
| *All words* | | | | |
| baseline | 0.629 | 0.474 | 0.079 | 0.834 |
| gpt_small | 0.620 | 0.489 | 0.078 | 0.836 |
| gpt_j | 0.624 | 0.487 | 0.079 | 0.834 |
| prom | 0.582 | 0.583 | 0.079 | 0.842 |
| *content words* | | | | |
| baseline | 0.630 | 0.492 | 0.067 | 0.871 |
| gpt_small | 0.622 | 0.502 | 0.066 | 0.871 |
| gpt_j | 0.625 | 0.501 | 0.067 | 0.870 |
| prom | 0.583 | 0.595 | 0.067 | 0.873 |
| *stop words* | | | | |
| baseline | 0.629 | 0.417 | 0.082 | 0.779 |
| gpt_small | 0.615 | 0.445 | 0.081 | 0.780 |
| gpt_j | 0.617 | 0.437 | 0.081 | 0.782 |
| prom | 0.580 | 0.544 | 0.081 | 0.793 |

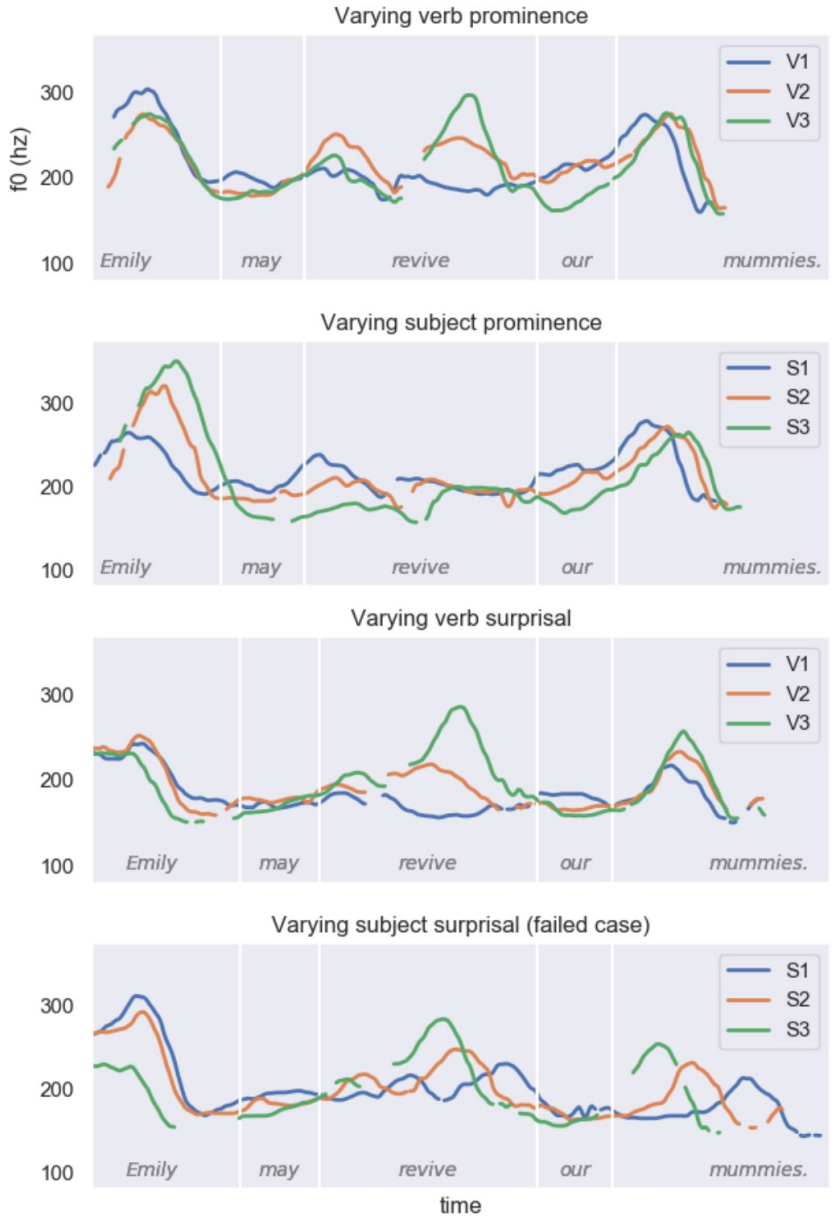



Figure 5: $f_0$ *trajectories of synthesized utterances illustrating the controllability of word prosody by prominence and surprisal conditioned systems. The numbers 1-3 refer to low, average and high prominece and surprisal, respectively.*

# Conclusions

- Prominence correlated with surprisal rather weakly, and increasing the model size and the context window had mostly detrimental effect.
- The synthesis results also suggest a limited efficacy of using surprisal values for eliciting appropriate prominence patterns, while providing a minimal improvement over the baseline.
- In general, information theoretical aspects explain only a part of the prosodic variation.

# B. Large Speech Models for prosodic analysis tasks

Kakouros, S. and Hiovain-Asikainen, K. (2023). North Sámi Dialect Identification with Self-supervised Speech Models. In proceedings of *the Annual Conference of the International Speech Communication Association (Interspeech-2023)*, Dublin, Ireland, pp. 5306–5310. 10.21437/Interspeech.2023-1928

# Introduction

- The North Sámi language (NS) encapsulates four primary dialectal variants.
- ~20,000 – 30.000 speakers
- NS is spoken in the northernmost parts of Norway, Sweden and Finland.
- NS speakers are in many cases bilingual in Sámi as well as in the dominant state language.
- ***Can we automatically identify between the NS dialects?***
- ***Are the NS dialects affected by the state language?***

# Background

- Dialect identification (DID) systems can be used to improve the performance of TTS and ASR systems.
- DID systems make use of the (i) text, (ii) speech, or (iii) text + speech of the dialects.
- DID typically performed using ML methods based on acoustic and/or textual features.
- Self-supervised speech models have not been used before in DID.

# Data

Table 1: ***Utterance*** *counts of the North Sámi recordings in their respective varieties based on the majority language.*

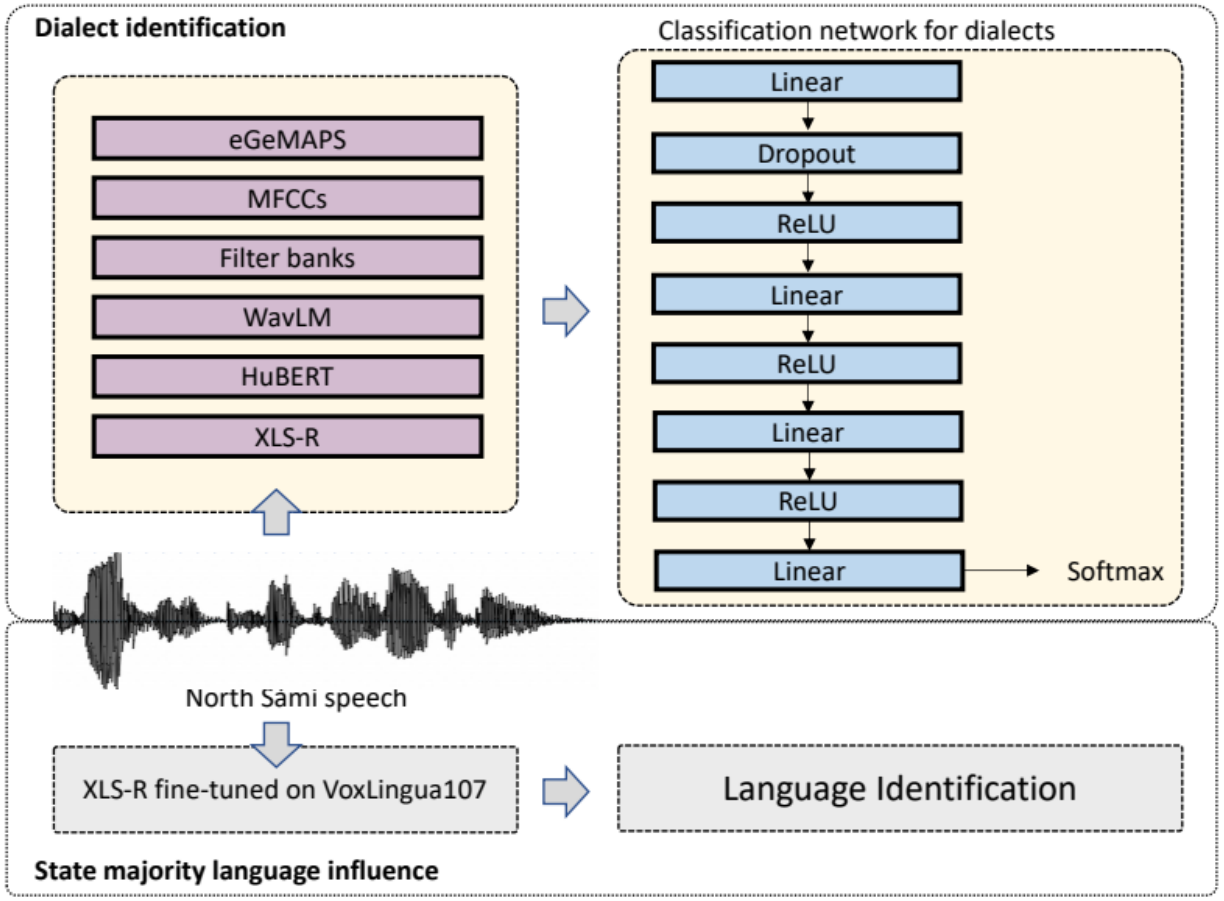
| *Dialect* | **Finnish** | | **Norwegian** | | **Swedish** | | *Total* |
|---|---|---|---|---|---|---|---|
| | *f* | *m* | *f* | *m* | *f* | *m* | */dialect* |
| *WF* | 21 | 38 | 4564 | 4943 | 0 | 0 | 9566 |
| *EF* | 1861 | 2864 | 835 | 3644 | 0 | 0 | 9204 |
| *SS* | 102 | 212 | 1398 | 2201 | 0 | 0 | 3913 |
| *TS* | 363 | 639 | 624 | 1602 | 63 | 166 | 3457 |
| *Total* | 2347 | 3753 | 7421 | 12390 | 63 | 166 | **26140** |

- Combined from four datasets:
  - LIA Sápmi North Sámi corpus
  - The Giellagas Corpus of Spoken Saami Languages
  - Divvun/Acapela TTS corpus
  - DigiSami read speech corpus
- Data consist of 30% read and 70% spontaneous speech.
- Data were split to utterances and labeled according to the dialect groups of the speakers as well as speakers' gender and country of origin.
- According to the traditional dialectological analysis of NS, the language can be divided to four main dialect groups: the Western Inland, the Eastern Inland, the Torne, and the Sea.

# Methods – architecture

- DID: classification task that includes a light-weight classification network using:
- Standard acoustic features such as MFCCs, filter banks, prosodic features.
- Self-supervised speech representations (WavLM, HuBERT, XLS-R).
- Majority language influence (MLI): experiment on language identification (LID):
- Spoken LID model that is fine-tuned using weights of the pretrained XLS-R on the VoxLingua107 corpus that includes Finnish, Norwegian, and Swedish.

# Methods – architecture



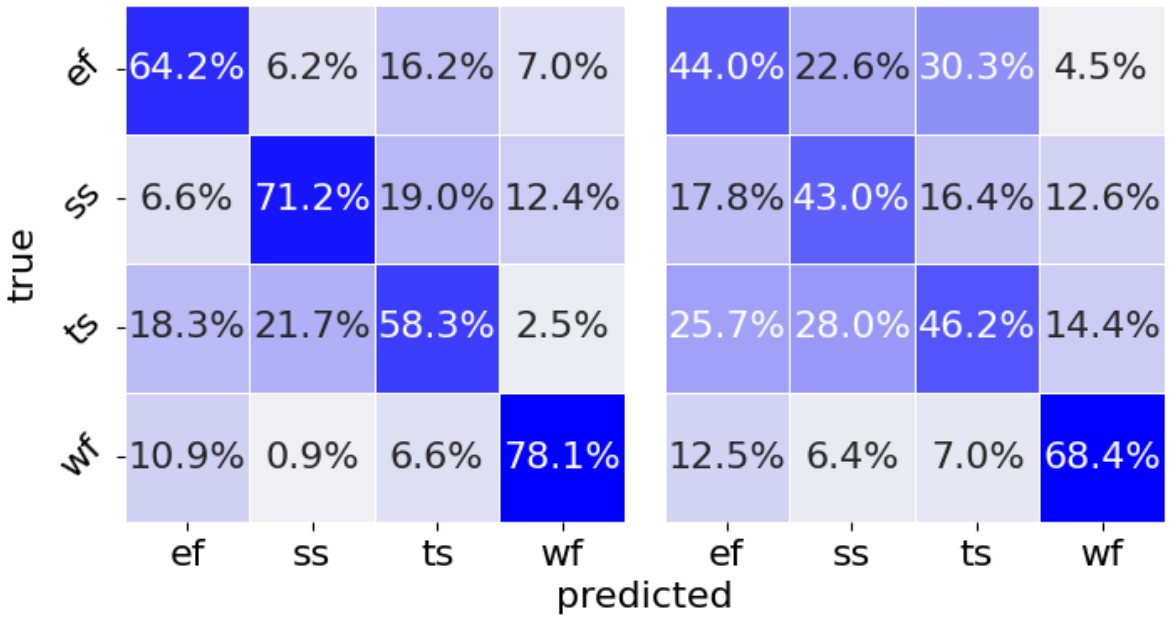
# Results – dialect identification

- Perform experiments in both speaker-independent (SI) and speaker-dependent (SD) setups and average across 3 splits.
- SD: vary the random seed and perform uniform sampling in the entire dataset with a ratio of 80% training and 20% for testing.
- SI: in each split we leave 3 speakers from each dialect out (a total of 12 unseen speakers in the test set).
- Self-supervised representations perform the best (for SI and meanstd, XLS-R 62.9%, and HuBERT 47.9%).
- Traditional acoustic features perform systematically with lower accuracy.

# Results – majority language influence

- LID applied on all utterances in our data.
- n-best results reported: for each n, we get the top n results from the classifier.
- State language where the dialects are spoken has an important impact on the NS dialects.
- For Finnish and Norwegian the LID task identifies the correct language (1-best) in 28.9% and 20.3% of the cases respectively.

# Results




CMs for a speaker-independent split across the four North Sami dialects using meanstd pooling with XLS-R (left) and HuBERT (right) and DNN for classification.

Table 2: *Unweighted accuracy (%) across dialects with speaker-dependent (SD) and speaker-independent (SI) splits. Results are averaged across runs. Values in bold are the highest scores per row per SI/SD.*

| *Feature* | **mean** | | **std** | | **meanstd** | |
|---|---|---|---|---|---|---|
| | *SD* | *SI* | *SD* | *SI* | *SD* | *SI* |
| *eGeMAPS* | 60.1 | **32.9** | 55.8 | 30.1 | **63.4** | 32.1 |
| *MFCCs* | 52.9 | 29.7 | **86.1** | **38.3** | 83.7 | 37.0 |
| *FBs* | 63.4 | 29.3 | 58.0 | 28.5 | **67.2** | **32.7** |
| *HuBERT* | **82.5** | 46.2 | 65.1 | 44.5 | 82.2 | **47.9** |
| *WavLM* | **72.7** | 41.0 | 65.9 | 42.6 | 70.6 | **44.0** |
| *XLS-R* | **95.0** | 62.8 | 88.5 | 56.9 | 90.1 | **62.9** |

Table 3: *Majority language influence based on n-best language identification results. All reported values are percentages.*

| *Majority* | **1-best** | | | **2-best** | | | **5-best** | | |
|---|---|---|---|---|---|---|---|---|---|
| *language* | *fi* | *no* | *sv* | *fi* | *no* | *sv* | *fi* | *no* | *sv* |
| *Finnish* | **28.9** | 12.9 | 7.7 | **39.2** | 30.2 | 14.2 | **56.3** | 51.5 | 32.3 |
| *Norwegian* | 10.3 | **20.3** | 13.4 | 22.6 | **36.8** | 24.4 | 44.5 | **62.9** | 50.1 |
| *Swedish* | 13.5 | **17.5** | 17.0 | 32.8 | **36.7** | 24.5 | **56.3** | 55.46 | 53.7 |

# Conclusions

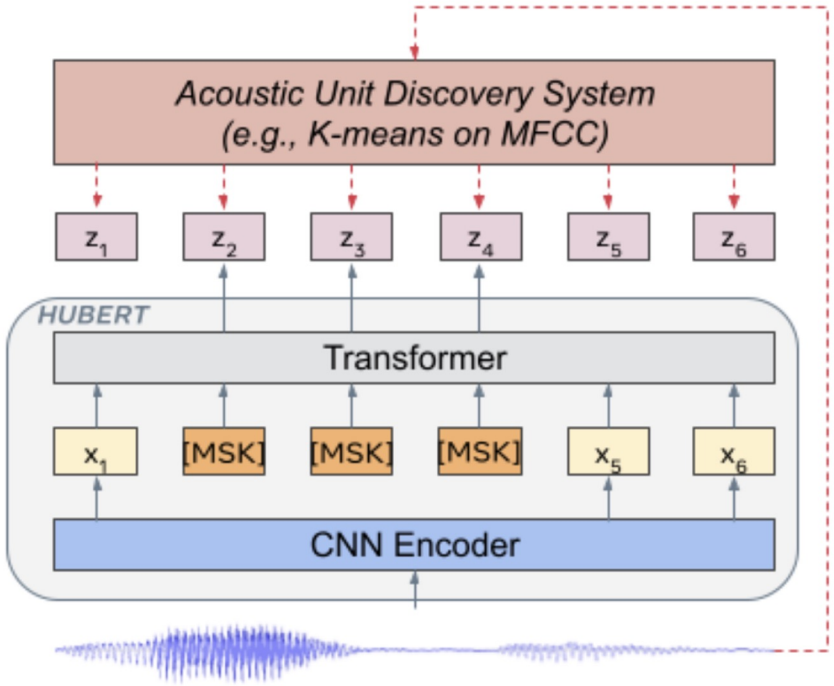
- We presented an extensive analysis of the classification potential of four NS dialects based on acoustic features.
- Our experiments are one of the first evaluating several self-supervised models in DID.
- We presented an approach for evaluating the majority language influence on dialects.
- We found the best performance in classifying the dialects using XLS-R and we also provide strong indications of the majority language influence on the dialects.

# C. Pooling methods from Large Speech Models for prosodic tasks

Stafylakis, T., Mošner, L., Kakouros, S., Plchot, O., Burget, L., and Černocky, J. (2023). Extracting speaker and emotion information from self-supervised speech models via channel-wise correlations. In *Proceedings of the IEEE Spoken Language Technology Workshop (SLT-2023)*, Doha, Qatar, pp. 1136–1143. 10.1109/SLT54892.2023.10023345

## Introduction

- Several self-supervised models for speech
  - e.g. Wav2Vec 2.0, HuBERT, WavLM
- Transformers as backbone network.
- Hard to finetune them for each single task.
- How well do they perform out-of-the-box?
  - e.g. in speaker and emotion recognition
- We may
  - add a trainable classifier,
  - use representations from all layers.




Training HuBERT

# SUPERB




**SUPERB**

**S**peech processing **U**niversal **PER**formance **B**enchmark

National Taiwan University 國立臺灣大學

Carnegie Mellon University

Massachusetts Institute of Technology

Johns Hopkins University

FACEBOOK AI

https://superbbenchmark.org

## SUPERB

### Backend models

- Speaker Identification (SID)
  - Average pooling + Classification head
- Speaker Verification (SV)
  - TDNN + Average pooling with std + Classification head (cosine similarity for evaluation)
- Emotion Recognition (ER)
  - Average pooling + Classification head

### Layer pooling

- Pool information from all layers based on a learnable set of weights
  - Different set of weights for each task
  - Weighting layers is a reasonable operation for nets with residual connections

$$\mathbf{h}_t = \sum_{l=0}^{L} \gamma_l \mathbf{h}_{t,l}$$

## Standard way of pooling frames

## Average pooling

- Too simplistic…
- The SSL networks are not trained to preserve speaker/emotion info.
- Their losses suppress this information in uppermost layer.
- Augmenting the pooled representation with standard deviation (std) helps.
- But can we do better?

$$\mathbf{r} = \bar{\mathbf{h}} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{h}_t$$

Average pooling

$$\mathbf{r} = \left[\bar{\mathbf{h}}; \left(\frac{1}{T}\sum_{t=1}^{T}(\mathbf{h}_t - \bar{\mathbf{h}})^2\right)^{1/2}\right]$$

Average pooling augmented with std

## Proposed way of pooling frames

## Correlation pooling

- What about modelling correlations between channels?
- We first reduce the #channels (e.g. to 128) using a learnable linear layer.
- We then standardize (i.e. mean = 0, std = 1) each channel.
  - For each utterance, not globally.
- Then, average pooling of the frame-wise outer products ⇒ correlation matrix:

$$\mathbf{C} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{o}_t \mathbf{o}_t'$$

- Dropout whole channels is a helpful regularizer.
- But do these correlation coefficients carry speaker/emotion information?

# Experiments in Speaker Recognition

**Table 1**. Results in SID and SV following the training and evaluation protocol defined by SUPERB.

| HuBERT Large | | |
|---|---|---|
| Pooling method | SID [acc. %] | SV [EER %] |
| mean | 89.3 | – |
| meanstd | 90.6 | 6.2 |
| correlation | 95.5 | 5.3 |
| corr. w/ dropout | 95.3 | 4.8 |
| mean+correlation | 96.2 | – |
| meanstd+correlation | 96.3 | 5.1 |
| meanstd+corr. w/ drop. | 96.3 | 4.8 |
| WavLM Large | | |
| Pooling method | SID | SV |
| mean | 93.0 | – |
| meanstd | 94.9 | 4.8 |
| correlation | 97.7 | 4.5 |
| corr. w/ dropout | 97.7 | 4.1 |
| mean+correlation | 97.7 | – |
| meanstd+correlation | 98.2 | 4.0 |
| meanstd+corr. w/ drop. | 98.6 | 3.8 |

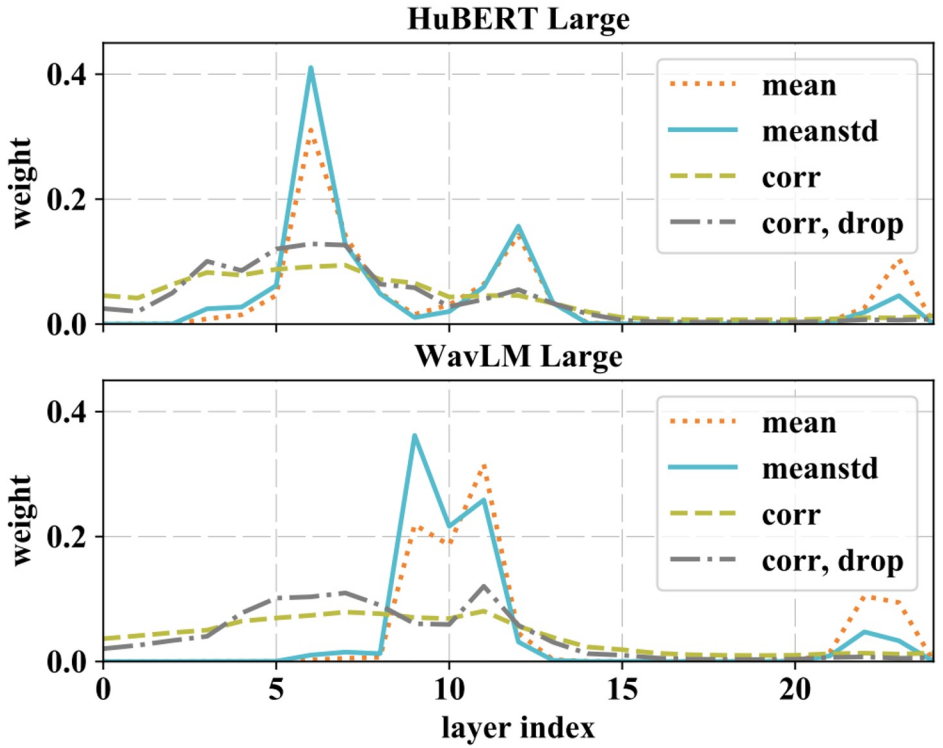



**Fig. 1**. Learned weights per HuBERT-Large and WavLM-Large layers for SID.

# Experiments in Emotion Recognition

**Table 2**. Unweighted accuracy (%) for the emotion recognition task in IEMOCAP using HuBERT and WavLM self-supervised representations.

| Pooling method | HuBERT Large | WavLM Large |
|---|---|---|
| mean | 60.46 | 66.73 |
| meanstd | 68.57 | 69.86 |
| correlation | 68.21 | 71.43 |
| corr. w/ dropout | 69.95 | 71.43 |
| mean+correlation | 68.94 | 71.43 |
| meanstd+correlation | 69.22 | 70.41 |

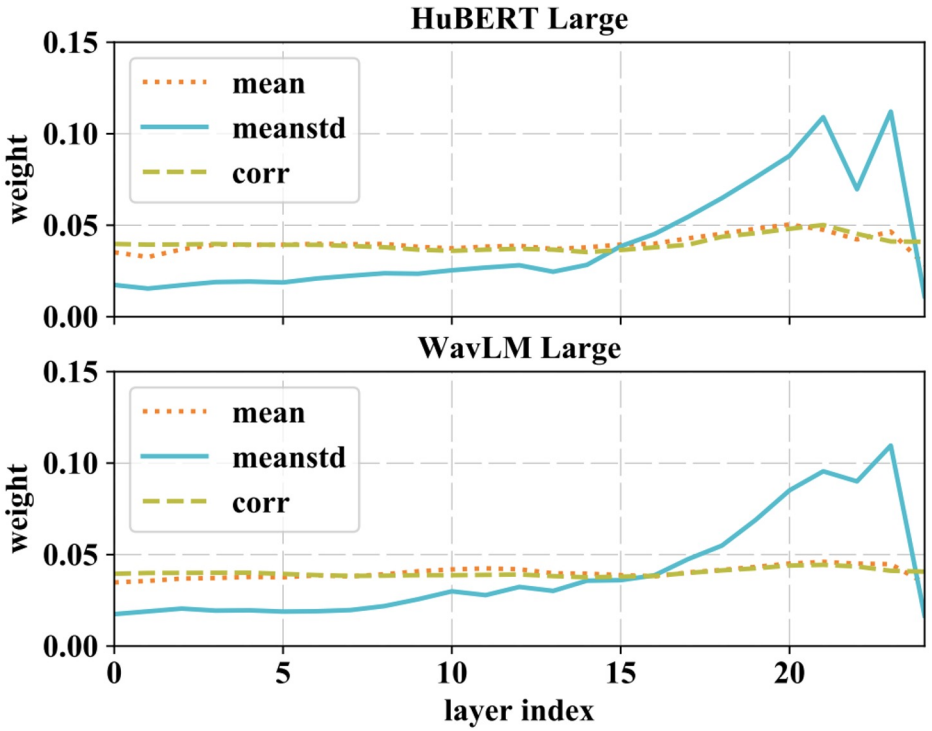



**Fig. 3**. Learned weights per HuBERT-Large and WavLM-Large layers for ER.

Kakouros, S., Stafylakis, T., Mošner, L., and Burget, L. (2023). Speech-based emotion recognition with self-supervised models using attentive channel-wise correlations and label smoothing. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-2023)*, Rhodes, Greece, pp. 1–5. 10.1109/ICASSP49357.2023.10094673

# Introduction

- Emotional expressions are a fundamental component of spoken interaction.
- However, recognizing emotions in speech remains a challenging problem:
  1. What kind of acoustic representations are best for speech emotion recognition?
  2. How can we best model the long temporal context over which emotions take place?
  3. How can we best tackle the problem of ambiguous labels for the emotions?

# Background

- Emotion recognition is typically treated as an utterance-level task.
- Frame-level pooling:
  - Mean
  - Standard deviation
  - Mean + standard deviation
- ***Can we do things better and capture more informative representations from the successive frames?***

# Method – correlation pooling

- Modelling correlations between channels.
- We first reduce the number of channels from 1024 to 256 using a learnable linear layer.
- Then, average pooling of the frame-wise outer products ⇒ correlation matrix:

$$\mathbf{C} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{o}_t \mathbf{o}_t'$$

- But emotion information does not appear uniformly across our signals. What can we do?

# Method – attentive correlation pooling

- We introduce a new flavour of attention by inserting weights in the estimates of the statistics:

$$\boldsymbol{\mu} = \sum_{t=1}^{T} w_t \mathbf{v}_t, \ \boldsymbol{\Sigma} = \sum_{t=1}^{T} w_t (\mathbf{v}_t - \boldsymbol{\mu})(\mathbf{v}_t - \boldsymbol{\mu})'$$

- The proposed attention enables us:
  - to keep the multi-modality of multi-head attention since a single head is too weak to capture the phonetic, speaker, emotion and channel variability,
  - to robustly estimate the attention by aggregating the matrix similarities prior to the Softmax function.
- ***However, our labels contain a certain degree of ambiguity that we need to address.***

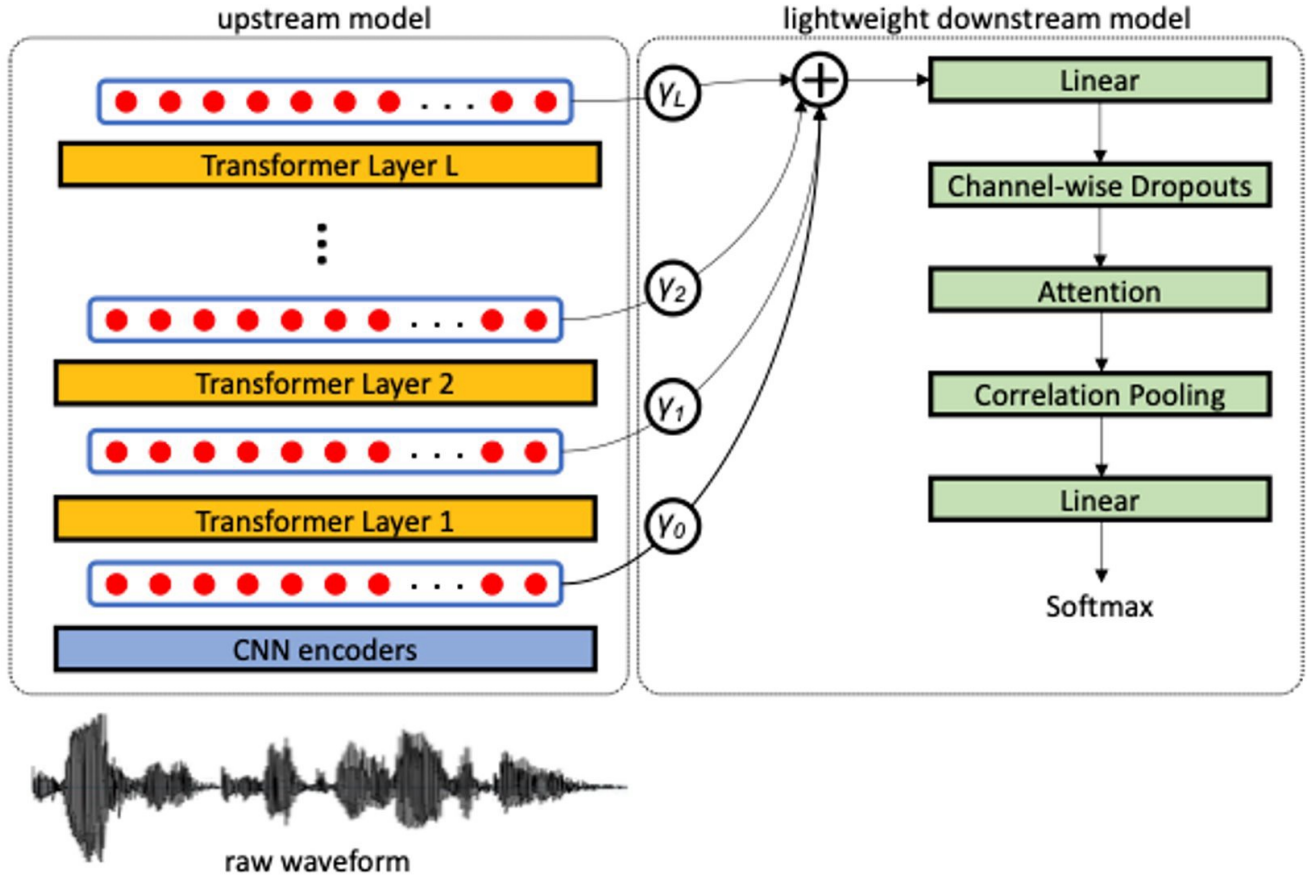
# Method – label smoothing

- With label smoothing we soften the hard (one-hot) targets vectors.
- The aim of label smoothing is to reduce the confidence of the classifier on the target labels.
- Label smoothing replaces the one-hot encoded labels with a mixture of the one-hot encoded labels and the uniform distribution.
  - One-hot encoded labels maximize logit gaps that are fed into the Softmax function.
  - On the other hand, smoothed labels, encourage smaller logit gaps, thus reducing the confidence for the targets.

Müller, R., Kornblith, S., & Hinton, G. E. (2019). When does label smoothing help?. *Advances in neural information processing systems*, *32*.

# Methods – architecture

- Our setup is based on SUPERB–
  Speech processing
  Universal PERformance Benchmark
- SUPERB is a collection of benchmarking resources to evaluate the capability of a universal shared representation for speech processing
- We extract embeddings from all transformer layers:
  - HuBERT
  - WavLM
  - Wav2Vec 2.0
- Layer pooling
- Weigh embeddings
- Channel-wise dropouts
- Apply attention
- Correlation pooling



# Results

- Experiments run on IEMOCAP
- 5-fold cross-validation
- Our method yields results that surpass those on the benchmark setup of SUPERB
- SUPERB reports an accuracy of 70.62% with WavLM and 67.62% with HuBERT
- With our proposed approach we obtain 75.60% and 73.86% respectively

**Table 1**. Unweighted accuracy (% mean and std) between test sets for SER in IEMOCAP using HuBERT large, Wav2vec 2.0, and WavLM large self-supervised representations.

| Pooling method | **HuBERT** | **Wav2vec 2.0** | **WavLM** |
|---|---|---|---|
| mean | 65.73 (2.73) | 66.86 (1.76) | 69.44 (1.53) |
| mean-std | 69.15 (1.61) | 69.92 (1.17) | 72.56 (1.67) |
| corr ($p_d = 0$) | 69.82 (1.35) | 68.44 (1.85) | 72.34 (1.54) |
| corr ($p_d = 0.25$) | 69.72 (1.19) | 67.85 (1.84) | 72.27 (1.45) |
| corr attentive | **73.86 (2.10)** | **70.01 (2.20)** | **75.60 (2.33)** |

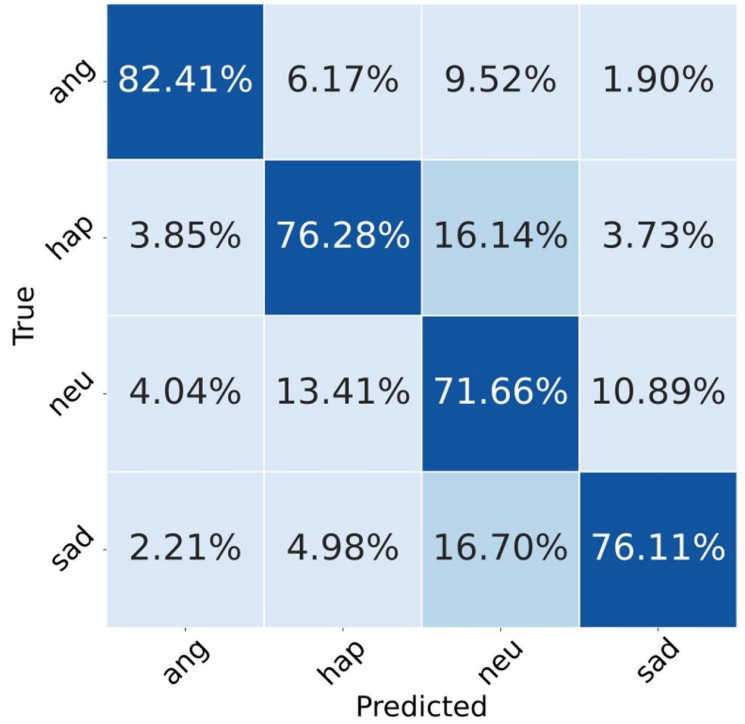



Figure: Attentive correlation pooling with WavLM

# Conclusions

- SER framework that uses self-supervised representations and is based on label smoothing and a novel approach to attention: attentive correlation pooling.
- Our method does not require fine-tuning of the pre-trained SSL models but rather uses a light-weight classification head.
- Our method reaches high performance in all pre-trained models tested surpassing that of the literature in similar tasks.

Thank you!