Improving end-to-end ASR systems using prosody-based curriculum learning

A dissertation submitted by

Ariadna Frutos

in partial fulfilment of the requirements
for the degree

Master in Language Technologies

Escuela Técnica Superior de Ingeniería Informática
Universidad Nacional de Educación a Distancia

Directed by

Mireia Farrús (UB)
Juan Manuel Cigarrán (UNED)

June 2022

Acknowledgments

I would like to express my deepest gratitude to the team that has contributed to this work: Mireia Farrús, Guillermo Cámbara, Jordi Luque and Juan Manuel Cigarrán. Thank you all.

Abstract

Curriculum learning (CL) is a machine learning technique that sorts the training examples according to their difficulty and exposes them to the learner progressively. In the field of automatic speech recognition (ASR) it has been successfully applied using different approaches, but none of them has taken advantage of prosody effects to weight the difficulty of data. In this work we present the first attempt to improve performance of end-to-end ASR systems making use of a CL strategy based on prosody. We start by evaluating transcriptions made by a pre-trained ASR model from a collection of utterances and compare the accuracy achieved, in terms of WER, with 18 scalar prosodic features extracted from each utterance. We find that transcriptions of utterances having either a more pronounced pitch or intensity contour achieve, in average, a lower WER. That is, these utterances are easier to transcribe. The standard deviation of the fundamental frequency shows the reverse behaviour: transcriptions of utterances having a large standard deviation achieve, in average, a higher WER. Then we train a new ASR system from scratch using a curriculum based on pitch contour. Our results indicate that curriculum learning based on this prosodic feature does help the system to learn but is not powerful enough to compensate for the negative effects of feeding the system gradually.

Resumen

El *Aprendizaje curricular* (o CL, del inglés *Curriculum Learning*) es una técnica de aprendizaje automático que ordena los ejemplos de entrenamiento según su grado de dificultad y los pasa al alumno de forma progresiva. En el campo del reconocimiento del inglés *Automatic Speech Recognition*) se ha aplicado con automática del habla (ASR, éxito siguiendo distintas aproximaciones, pero ninguna ha aprovechado efectos prosódicos para ponderar la dificultad de la señal vocal. En este trabajo presentamos el primer intento de mejorar el rendimiento de sistemas ASR de extremo a extremo utilizando una estrategia de CL basada en prosodia. Empezamos evaluando las transcripciones realizadas por un sistema ASR pre-entrenado para una colección de declaraciones, y comparamos la precisión alcanzada, en términos de WER, con 18 parámetros prosódicos extraídos de cada declaración. Los resultados muestran que las transcripciones de declaraciones que tienen un contorno de tono o intensidad más pronunciado tienen, en promedio, una WER menor. Es decir, este tipo de declaración es más fácil de transcribir. La desviación estándar de la frecuencia fundamental muestra el comportamiento opuesto: las transcripciones de declaraciones con una desviación estándar elevada tienen, en promedio, una WER mayor. En la segunda parte de la investigación entrenamos un nuevo sistema ASR desde cero utilizando un currículum basado en el contorno de tono. Los resultados obtenidos indican que la aplicación de una estrategia de CL basada en este parámetro prosódico ayuda al sistema a aprender, pero no es lo suficientemente potente como para compensar los efectos negativos que provoca el hecho de alimentar el sistema de forma gradual.

Table of contents

Figure Index

Table Index

1 Introduction

Automatic Speech Recognition (ASR) is the technology that allows computers to transcribe spoken language into text. It is the key piece of voice interfaces, which allow the introduction of simple information into a system using our voice instead of a keyboard. Domestic voice-enabled virtual assistants are becoming more and more popular thanks to the efficiency and speed they offer to perform tasks such as keyword search, phone dialling, data entry, or home appliances' control. In the industry, voice interfaces are being used to control robots, perform computer commands, or introduce simple information to a database when hands and eyes are busy. As for commercial applications, voice interfaces are nowadays the front gate in most call centres. An even more challenging application is dictation of reports.

State-of-the-art ASR systems are based in artificial intelligence. Recently, the so-called "end-to-end" models are picking up steam, not only in research but also in production settings. These systems use a neural network-based model that directly maps sequences of input acoustic features into sequences of words, without the need of an acoustic model, lexicon, nor language model. As any other neural network-based architecture, this kind of system needs to be trained to find the appropriate weights of the network and achieve a correct mapping from inputs to outputs. Training requires having huge datasets, along with substantial computer resources, and it is very time-consuming. Even with all this, accuracies are often far from perfect. Improving the model performance and accelerating the training process with smaller datasets are major objectives in machine learning research. The field of ASR is not an exception.

A new technique that is being explored to train neural networks is *Curriculum learning* (CL). The term refers to a general approach to optimize the learning process of a machine learning system. The basic idea is to start small, learn easier aspects of the task or easier sub-tasks, and then gradually increase the difficulty level. The method is inspired by human teaching, where concepts are not introduced randomly but organized in a meaningful order that illustrates gradually more concepts, and more complex ones.

The power of CL has been exploited in a wide range of supervised learning tasks, including computer vision, natural language processing, healthcare prediction and graph learning tasks. The application of this technique to improve the training time and final transcription accuracy of ASR systems has also been addressed. A successful approach has been to consider the length of utterance as a measure of difficulty: the longer the utterance, the more difficult to transcribe. CL strategies based on this hypothesis apply training schedules in which the system is first trained with short utterances and fed with longer ones in later epochs. Alternative features that have been postulated to define “easy” and “difficulty”, have been noise and word rarity.

As far of we know, prosody, i.e., the patterns of stress and intonation, has not been used as a measure of difficulty for CL strategies. Yet prosody provides humans with important markers both at word and syntactic level. Compare the stress in the word “present”, the noun (/ˈprez.ᵊnt/), with that of “present”, the verb (/prɪˈzent/), and see the difference between the intonations of the sentence “I’d like to eat, mummy” and “I’d like to eat mummy”. Not only for people is speech easier to understand if pronounced with a suitable intonation; some studies have shown that accuracy in speech transcription made by machines is also related with certain prosodic features. On the other hand, there is evidence that babies are sensitive to prosodic markers of syntactic units, and they use this sensitivity to recognize phrasal units, both noun and verb phrases, in fluent speech.

All this prompts us to pose the following question: could prosodic features be suitable candidates to define CL strategies that improve the training of ASR systems? This work is a preliminary attempt to answer this question. In particular, we intend to find prosodic features that show a relationship with the accuracy at which an ASR system transcribes speech, and use them in experiments that point the way forward for future research to improve performance of ASR systems using CL.

1.1 Motivation

The idea of organizing training data into a schedule, or “curriculum”, based on the easiness or difficulty of the samples was formalized and popularized by (Bengio et al., 2009). In this work, several experiments are presented, involving vision and natural language tasks, which show that pre-training with a CL strategy can improve generalization and help to reach faster convergence. The vision task is very illustrative of the method. Its final objective is to train a system in the classification of geometrical shapes into 3 classes: rectangle, ellipse, and triangle. To this purpose, two datasets are generated: a “difficult” one, which includes all kind of figures, and an “easy” one, which

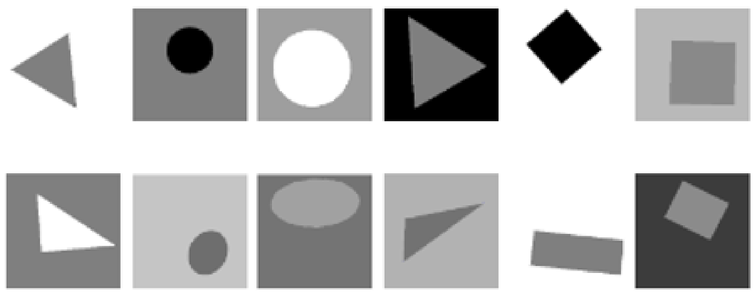


Figure 1. Sample inputs from the “easy” (top) and “difficult” (bottom) datasets in the vision experiment by (Bengio et al., 2009). Source: (Bengio et al., 2009)

only includes squares, circles, and equilateral triangles. An illustration of both datasets is shown in Figure 1. The curriculum consists of a 2-step schedule: first, the system is trained using the easy training set during a predefined number of epochs; then the training continues using the difficult set. Experiments made using this 2-step schedule outperformed the baseline, which corresponded to training only on the difficult set.

Curriculum learning has open new avenues of research in the optimization of neural networks. If well designed, CL strategies can significantly improve the model performance and accelerate the training process, two of the most desired objectives in machine learning research. Technically, it can be implemented as an independent plug-and-play module, what makes it easy to use.

In ASR, different approaches have been successfully tested, but none that uses prosody as a measure of difficulty. Prosody refers to those properties of speech that can be perceived when we listen to segments larger than phonemes, such as syllables, words, and phrases. Examples of prosodic features are pitch, loudness, intonation, rhythm, voice quality, pauses, speech rate and duration. Altogether, these features provide the effect of *melody*. Prosodic boundaries represent a significant cue to recognize lexical candidates in a sentence (Salverda et al., 2003), and human comprehension, in terms of cognitive load and accuracy, is better when we listen to natural prosody, as opposed to monotone or foreign prosody where word stress is missing or not as expected (Hahn, 1999). On the other hand, there is evidence that infants as young as 6 months are sensitive to prosodic markers of syntactic units and use them to recognize phrasal units in fluent speech (Soderstrom et al., 2003), which suggests that prosody may be instrumental in human language acquisition. If prosody plays a role in helping humans to understand speech and babies to learn a language, it seems reasonable to believe that it might be important for machines as well. Training of ASR systems may consequently benefit from CL strategies based on prosodic features.

1.2 Objectives

Given the importance of prosody in speech comprehension and language acquisition by humans, it makes then sense to hypothesize that i) it might be easier/more difficult for ASR systems to transcribe utterances that are modulated in a certain way; and ii) prosodic features that make utterances easy/difficult to be transcribed are feasible candidates to define CL strategies that improve performance of ASR systems.

In this work we intend to shed some light on these hypotheses. Concerning the first hypothesis, we will assume that an utterance is "easy" if the transcription provided by a pre-trained ASR system is accurate in terms of the number of errors. That is, the fewer the errors of the transcription, the easier the utterance. We are interested in exploring the relationship between certain prosodic features and the easiness to transcribe speech.

For instance: are utterances pronounced at a higher pitch easier to transcribe than utterances pronounced at a lower pitch? Does the direction of the intonation contour have some effect? Do features like harmonics-to-noise ratio, jitter and shimmer, which are related to the quality of speech, have any influence?

With respect to the second hypothesis, our concern is to examine up to which point prosodic features can be useful to train ASR systems using CL strategies. Does the application of a schedule in which the system is first trained with utterances having certain prosodic features improve training somehow? To face this question, we will train a new ASR model from scratch organizing the training data into a schedule, or “curriculum”, based on the previous results.

1.3 Structure of this document

The structure of the document is as follows. In Section 2, “Related work”, we provide a general review on CL and include our own survey of studies that have applied CL to end-to-end ASR systems. We also present several works showing that prosodic features play a role in speech recognition and related fields. In Section 3, “Case study”, we explain the procedure that has been followed to carry out our experiments; introduce important concepts related to prosody; describe the 18 prosodic features that will be analysed, as well as the evaluation metrics that will be used; and give detail of the experimental set-up. In section 4, “Experiments and results”, we provide insight of the experiments that have been conducted and report results and their interpretation. Finally, in section 5, “Conclusions and further work”, we summarize the work done and results achieved, draw conclusions, and suggest ways of improvement.

2 Related work

Our proposal combines two concepts that have been studied independently and have never, to our knowledge, been applied together, i.e., the advantages of training neural networks using curriculum learning strategies and the influence of prosodic features in automatic speech recognition. The following sections present an overview of the state-of-the art of each of these lines of research.

2.1 Curriculum learning

Although there are previous studies on the idea of introducing a curriculum into the training strategy of machine learning algorithm, such as (Elman, 1993), it was (Bengio et al., 2009) who formalized and popularized it. This work hypothesizes that a well-chosen curriculum strategy can both i) help to find better local minima of a non-convex training criterion and ii) accelerate training because the learner wastes less time with noisy or harder to predict examples. The work also illustrates the potential power of the method through several experiments where simple curriculums are tested, such as the one presented in section 1.1.

Since (Bengio et al., 2009), researchers have been exploiting power of CL in a wide range of applications. The paradigm has also been extended to modified versions within the spirit of "training from easier to difficult". In the two following sections we introduce the general frameworks that have been adopted under the paradigm "curriculum learning", following the classification by (X. Wang et al., 2020), and present our own survey of the state-of-the art in the field of ASR.

2.1.1 Curriculum learning approaches

(X. Wang et al., 2020) classifies the existing CL approaches in three types, depending on what is understood for "curriculum":

1. **Original Curriculum Learning**: A curriculum is a sequence of training criteria over T training steps, $C = \langle Q_1,...,Q_t,...,Q_T \rangle$. Each criterion Q_T is a reweighting of the target training distribution $P(z)$:

$$Q_t(z) \propto W_t(z) P(z) \ \forall \text{ example } z \in \text{training set } D.$$

such that the following three conditions are satisfied:

1- The entropy of distributions increases: $H(Q_t) < H(Q_t+1)$ (this means that the diversity and quantity of information of the training must increase)

2- The weight for any example stays the same or increases: $W_t(z) \leq W_{t+1}(z)\ \forall\ z \in D$ (weights of individual examples need to increase so that they can get added into the training set).
3- The final criterion matches the complete training distribution: $Q_T(z) = P(z)$.

2. **Data-level Generalized Curriculum**: A curriculum is a sequence of reweighting of target training distribution over T training steps. This version keeps condition (1) while relaxing conditions (2) and (3) in order to enable CL strategies that change the task from during training. The philosophy "training from easy to difficult" remains in the difficulty of the current task, which gradually increases until reaching the target task.

3: **Generalized Curriculum Learning**: A curriculum is a sequence of training criteria over T training steps. Each criterion Q_t includes the design for all the elements in training a machine learning model, such as tasks, model capacity, learning objective or loss function.

There also exist different approaches to define the difficulty metric and schedule. Pre-defined methods design them using human prior knowledge. The difficulty metric is usually based on the data characteristics of specific task, and it is designed considering complexity, diversity, and/or noise estimation. Some examples of difficulty metrics used in literature are regularity in shape (Bengio et al., 2009), sequence length, number of objects in images; number of coordinating conjunctions or phrases, information entropy, signal to noise ratio, image intensity and human-annotation-based Image Difficulty Scores. Concerning the schedule, we can classify them into two types: discrete schedules, which add new data after a fixed number of epochs or convergence value, and continuous schedules, which add training data subset at every epoch following a function $\lambda(t)$, where $\lambda(t)$ is the proportion of easier examples to be included in the t-th epoch.

Several successful attempts have also been made using automatic methods, where the curriculum is generated dynamically using data-driven models or algorithms. Some of these methods are:

- **Self-Paced Learning (SPL)** methods, which let the model himself act as the teacher and measure the difficulty of training examples according to its losses on them.
- **Transfer Teacher methods**, which use a pre-trained teacher model to measure the difficulty of training examples.
- **Reinforcement learning (RL)**, where a teacher makes dynamic data selection according to the feedback from the student.

2.1.2 Curriculum learning in end-to-end ASR systems

State-of-the-art ASR systems are mostly based on neural networks. One of the most successful architectures is the Deep Neural Network – Hidden Markov Model (DNN-HMM), a hybrid system composed of an acoustic, a lexicon, and language model. Most recently, though, end-to-end systems have reached state-of-the-art performance and are picking up steam, not only in research but also in production settings (Sainath et al., 2020). Figure 2 compares these architectures. End-to-end systems directly map sequences of input acoustic features into sequences of words. Common ASR end-to-end structures are Recurrent Neural Network Transducers (RNNT) (Graves, 2012), Connectionist Temporal Classification (CTC)-based systems (Graves et al., 2013; Hannun et al., 2014), and attention-based models, such as Google's Listen, Append, Spell (LAS) model (Chan et al., 2016) and Fairseq S2T (C. Wang, Tang, et al., 2020).

Besides their conceptual simplicity, end-to-end systems offer some important advantages. To begin with, the use of a single model, instead of three, makes them easier to train. They are also less expensive to build and require less human labour because there is no need of audio-word alignment nor of lexicon maintenance. Finally, they also bring important benefits when it comes to production: they are smaller in size, making them attractive for on-device ASR applications, and easier to optimize.

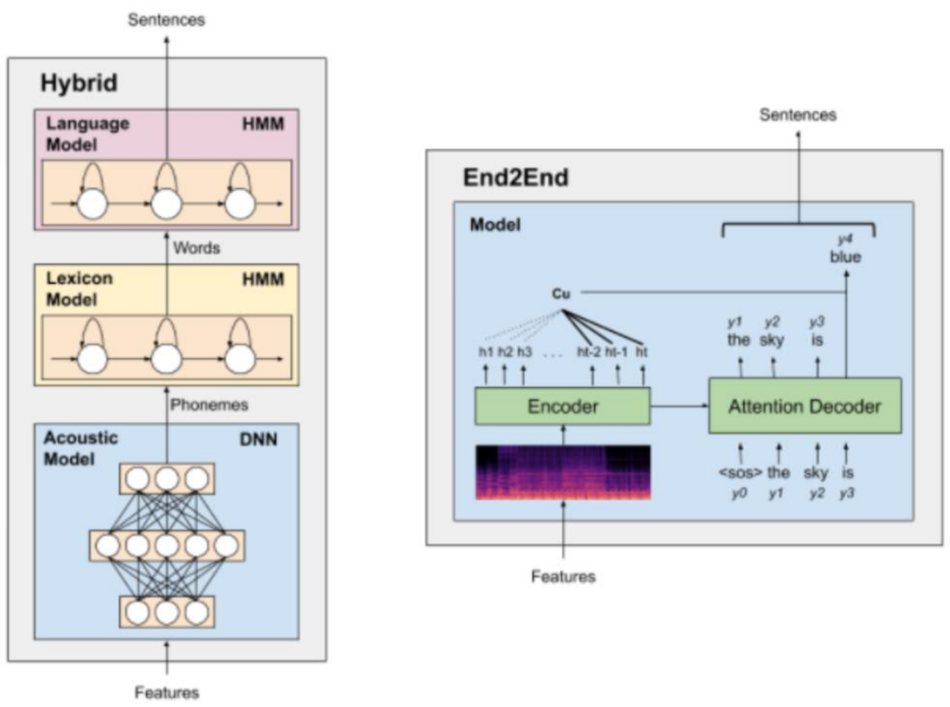


Figure 2. Comparison between a standard hybrid ASR architecture, composed of an acoustic, a lexicon, and language model, and a sequence-to-sequence based end-to-end model. Font: https://www.verbio.com/e2e-architectures-for-asr-systems/

There have been several successful attempts to train end-to-end ASR systems using CL strategies. In the following sections we present some of the approaches that have been explored, following the same classification as in Section 2.1.1.

2.1.2.1 Approaches using the original CL strategy

The mostly used scoring parameter for CL in ASR systems has been, by far, the length of the input sequences. For instance, in (Amodei et al., 2016) a module called SortaGrad is designed which applies a CL strategy to train a RNN based ASR system for English and Mandarin. The need of such strategy is motivated by the use of a CTC loss function, which tends to be unstable. The strategy consists in iterating, during the first epoch, through minibatches of the training set in increasing order of length of the longest utterance in the minibatch. SortaGrad improves the stability of training and achieves around 10% improvement in WER. On the other hand, (Kim et al., 2018) achieves a 7.8% WER decrease when training CTC-based speech recognition system by training it first on a subset that consists of shorter length utterances. Once these are learnt, they introduce the full training set. In (Zhang, Chang, Qian, & Watanabe, 2020) an end-to-end model for multi-speaker speech recognition is trained applying a CL strategy based on the length of the sequence during the first 3 epochs. During these epochs, the iteration of minibatches within the training set is made in ascending order of sequence length; after that, usual fine-tuning with random order of minibatches is done. They achieve a 10% improvement in WER. (Isik et al., 2016) uses a similar approach, but instead of the number of words they use the number of frames. In this case, the aim is to improve the performance of a multi-speaker separation system based on a deep clustering architecture. The study shows that pre-training with shorter segments followed by training with longer segments boosts performance from 9.9 dB to 10.3 dB for two speaker separation.

The second mostly studied scoring parameter in ASR is the signal-to-noise ratio (SNR, the energy ratio between the target speech and the interfering sources). This approach is used, above all, to improve speaker recognition in multi-speaker systems. The signal-to-noise ratio has a great influence on the final recognition performance in multi-speaker systems (Chang et al., 2019). When the speech energy levels for speakers in a conversation are markedly different, the recognition accuracy of those having lower levels is very poor. Following this criterium, (Zhang, Chang, Qian, & Watanabe. Shinji, 2020) and (Isik et al., 2016) investigate ascending SNR as a score function using analogue strategies as the ones described above for length-based strategies. They achieve improvements of 10% and 4% in WER, respectively. Similarly, (Chang et al., 2019) achieve a 12% improvement when training MIMO-Speech, their end-to-end multi-channel multi-speaker speech recognition proposed system, after applying a CL strategy that combines ascending sequence length and ascending SNR.

The use of SNR as a score function to train single-speaker ASR systems dealing with noise has been investigated by (Braun et al., 2017). In their experiments, two training schedules are proposed: one which starts training the network on the lowest SNR (noisier) samples and expands to high SNR samples in 5 dB steps; and the opposite one, i.e., a schedule expanding from high SNR to low SNR. Contrary to what could be expected, the best performance is achieved by the first schedule, which corresponds to the "anti-curriculum" strategy (noisy examples first). The authors consider that this result shows that noise allows the network to explore the parameter space more extensively at the beginning. Research using near-field and far-field data as a measure of difficulty, which is an indirect measure of noise (near-field data is less noisy) has been carried out by (Ranjan & Hansen, 2021). They obtain opposite results to (Braun et al., 2017): the proposed CL strategy, which consists in initiating the training with comparatively easier near-field data and including more diverse far-field data progressively in later stages, outperforms the baseline ASR system with relative reductions in WERs of up 10.1%.

Another feature used to define CL strategies in ASR has been the posterior probabilities obtained using a previously trained model (Vydana et al., 2016). Here, they start by training the network on the full training set. Then, the trained network is used to get the predictions on the same training data. The examples that are correctly predicted are considered as the easily examples. Then, they train the system again, now in two steps: initially, with a set of easy examples, and afterwards with the full training set. The model trained with proposed two-step CL learning strategy achieves a higher accuracy in a smaller number of epochs.

Gender has also been used as a score function in some CL strategies to train multi-speaker ASR systems (Zhang, Chang, Qian, & Watanabe. Shinji, 2020) and (Isik et al., 2016). Compression ratio of audio files has been tested by (Kuznetsova et al., 2021). In this case, the relevant improvement was not in precision, but in training speed. Finally, in speech-to-text translation, word rarity has been used in (Platanios et al., 2019).

2.1.2.2 Approaches using Data-Level Generalized Curriculum

A curriculum learning strategy with varying tasks to train an end-to-end ASR system has been investigated by (Kim et al., 2018). In their proposal, first, they reduce the number of categories to four symbols: vowel, consonant, space, and blank. As the simpler classification task is learned, the full character-based label set is restored and training proceeds. This strategy improved performance but did not work as effectively as one the that focused on short utterances, so no detailed results are provided.

2.1.2.3 Generalized Curriculum Learning

Attempts of application of curriculum learning to the architecture of the system, have been made in speech-to-text translation, a field closely related to ASR. In (Kano et al., 2017), the authors build an end-to-end attention-based speech-to-text translation system on syntactically distant language pairs. The authors build the system upon a standard attention-based encoder-decoder network that consists of an encoder, a decoder, and attention modules. They propose a structured-based curriculum learning strategy that consist in training the attentional encoder-decoder architecture by starting from a simpler task, switch a certain part of the structure (encoder or decoder), and set it to a more difficult task. Experimental results demonstrate that the learning model is stable and that the final translation quality outperforms that of the standard system. In (C. Wang, Wu, et al., 2020) a curriculum pre-training method is proposed to improve the power of the encoder in an end-to-end speech translation system. Traditional methods pre-train the encoder on ASR data to capture speech features and learn the alignment between the acoustic features and phonemes or words. In order to teach the model to understand the sentence and incorporate the required knowledge as well, the authors add a second pre-training phase with two extra courses: a frame-based masked language model (FMLM) task and a frame-based bilingual lexicon translation (FBLT) task. The proposed method significantly improves the "transformer + ASR" pre-train" baseline. They also pre-train the model on ASR, FMLM and FBLT tasks in one phase. The result is worse than the "transformer + ASR" pre-train" baseline, showing the importance of the curriculum learning strategy.

2.1.2.4 Automatic methods

As for automatic methods, (Kim et al., 2017) include in their paper a teacher-student technique to initialize the training of their CTC-based speech recognition system. Here, knowledge is transferred from a BLSTM offline model (teacher) to the final LSTM online system (student), achieving a 12.2% improvement on WER over the baseline. When combining this strategy with manual CL based on utterance length, the improvement increases up to 19.0%.

2.2 The effect of prosody aspects in automatic speech recognition

In the previous chapter we have mentioned some features of speech that have an influence in the average precision in which ASR and speech-to-text translation systems can perform, such as the length of the sequence, the signal-to-noise ratio or word rarity. Previous studies on these effects motivated the authors of the presented works to undertake corresponding CL strategies. Factors related to prosody have also been shown to be related with the difficulty experienced by ASR systems to recognize speech. In a dedicated study on the relationship between the errors of two ASR systems and several prosodic, lexical, and disfluency factors (Goldwater et al., 2010) showed that, in general, prosodic features are strongly predictive of error rates. The prosodic features included in their work were pitch (minimum, maximum and mean), intensity (minimum, maximum, mean and range), speech rate (phones per second), duration and log jitter. Pitch, intensity, and log jitter were extracted at word level; speech rate was computed at utterance level and assigned to all words in the utterance. The analyses on those features showed that precision decreased dramatically for fast speech. Mean pitch also had a large effect, with higher error rates for words with higher pitch relative to gender averages. Words with smaller ranges of pitch or intensity were more likely to be misrecognized, as were words with higher minimum intensity. Jitter and intensity maximum were associated with higher error rates at extreme values.

In fact, some prosodic features are already available in some ASR toolkits, such as Kaldi (Povey et al., 2011). In 2014 Kaldi introduced new algorithms to the framework to extract features for pitch, delta pitch and voicing (Ghahremani et al., 2014). These features can be appended to the raw input vectors (MFCCs or PLPs) before being forwarded to the classifier. In an experimental set-up, the addition of these prosodic features improved performance in 6% on tonal languages (Vietnamese and Cantonese) on 2% on atonal languages (Assamese and Bengali).

On the other hand, jitter and shimmer have been shown to play a role in several speech related tasks. Some examples are detection of speaking styles (Salverda et al., 2003; Slyh et al., 2008), age and gender classification (Wittig & Uller, 2003), emotion detection (Li et al., 2007) and speaker recognition (Farrús et al., 2007).

3 Case study

The case study proposed in this work attempts, firstly, to find feasible candidates among different kinds of prosodic features to define a CL strategy that helps improving transcription accuracy in ASR systems; secondly, to test candidate prosodic features in some experiments by training a new ASR model using the proposed CL strategy.

In the following sections we describe the general procedure that has been followed, the prosodic features that have been included in the analysis, the evaluation methods, and the experimental set-up.

3.1 General procedure

Our research will start by analysing the relationship between the value of some prosodic features for a collection of utterances and the accuracy that a trained state-of-the-art ASR model achieves when transcribing these utterances. We will call this system ASR1. Prosodic features showing a clearer relationship with the accuracy achieved by ASR1 will become our candidates to design a CL strategy. Later, we will undertake some curriculum learning experiments. We will design a CL strategy based on one of the previous candidates and we will train a new ASR model from scratch using this strategy. We will refer to the model trained using the CL strategy as ASR2. The final goal of this experiment will be to evaluate if such training has any effect on the final performance of the model.

3.1.1 Selection of candidate prosodic features

Prosodic features are those features of speech that contribute toward acoustic and rhythmic effects. In this work we will consider 18 different features related with pitch, intensity, and voice quality. All of them are scalars, computed as the average along the utterance being analysed. The complete list of features is included in section 3.2, along with an introductory description of their meaning. Among the list of prosodic features considered, the selection of candidates to define CL strategies will made by comparing their value for each utterance in a dataset collection and the accuracy achieved by ASR1, a pre-trained ASR system, in the transcription of those utterances.

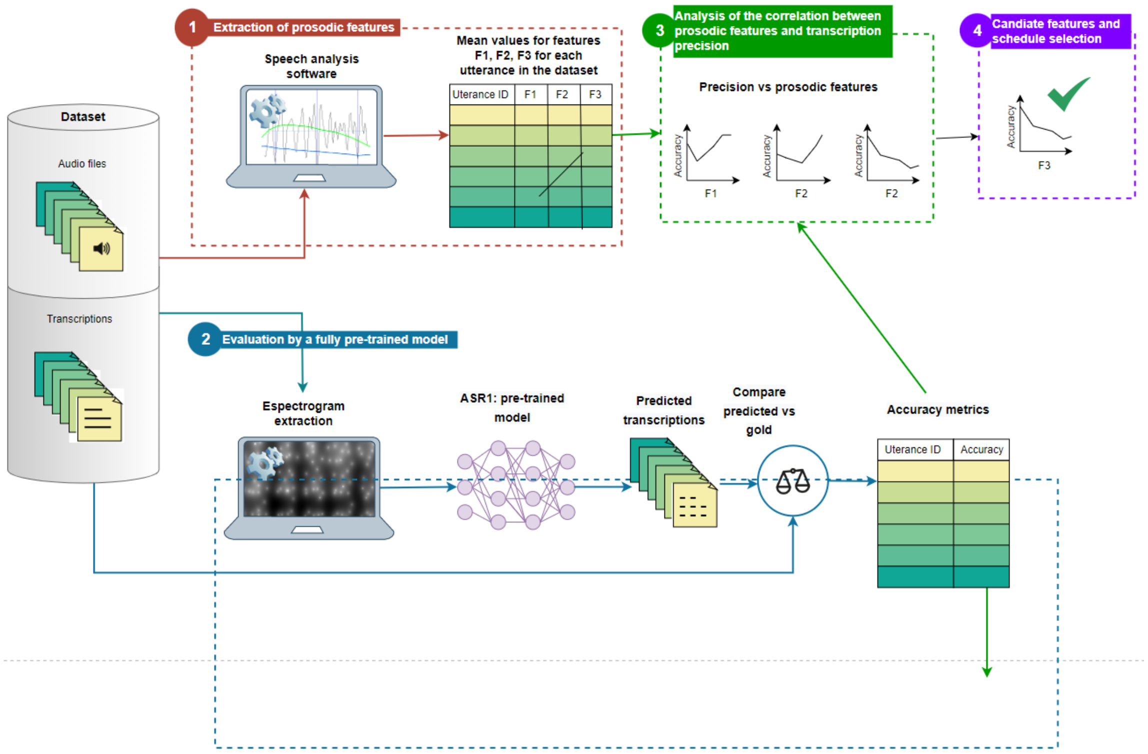

1 Extraction of prosodic features
Speech analysis software
Mean values for features F1, F2, F3 for each utterance in the dataset
Utterance ID
F1
F2
F3
3 Analysis of the correlation between prosodic features and transcription precision
Precision vs prosodic features
Accuracy
F1
F2
4 Candiate features and schedule selection
F3
Dataset
Audio files
Transcriptions
2 Evaluation by a fully pre-trained model
Espectrogram extraction
ASR1: pre-trained model
Predicted transcriptions
Compare predicted vs gold
Accuracy metrics
Accuracy

Figure 3. Procedure followed to select candidate prosodic features for a CL strategy.

Figure 3 shows the steps that will be followed. These steps are:

1- **Extraction of prosodic features:** For each audio file in the dataset, values for each prosodic feature will be extracted using a dedicated speech analysis software.

2- **Evaluation by ASR1:** A sequence modelling toolkit with a pre-trained model, ASR1, will be used to evaluate the dataset. The inputs to the model will be log-mel spectrograms extracted from the raw audio files from the dataset, each containing a speech utterance. As a result of this evaluation, each utterance will be assigned a score based on the number of errors of the transcription. The lower the accuracy, the easier an utterance will be considered.

3- **Analysis of the correlation between prosodic features and transcription precision:** Values for each prosodic feature from step 1 will be plotted against transcription accuracies from step 2 following different criteria.

4- **Selection of a difficulty metric:** Features showing a clearer relationship with transcription accuracy will be selected as candidates for a difficulty metric.

3.1.2 Curriculum learning experiments

The goal of this part of the research is to evaluate a CL strategy by training a new ASR system, ASR2, from scratch. This strategy will be defined taking into account the results of the first part of the project and applied using an independent module appended to ASR2. Figure 4 shows the architecture of the training system, which includes:

1. **The curriculum learning module**: This module takes the original dataset as input and organizes the data according to the curriculum. Then, it feeds the model gradually, following the schedule determined by curriculum.
2. **The ASR model**: This includes a state-of-the-art end-to-end ASR model and the necessary tools to train it and evaluate it.

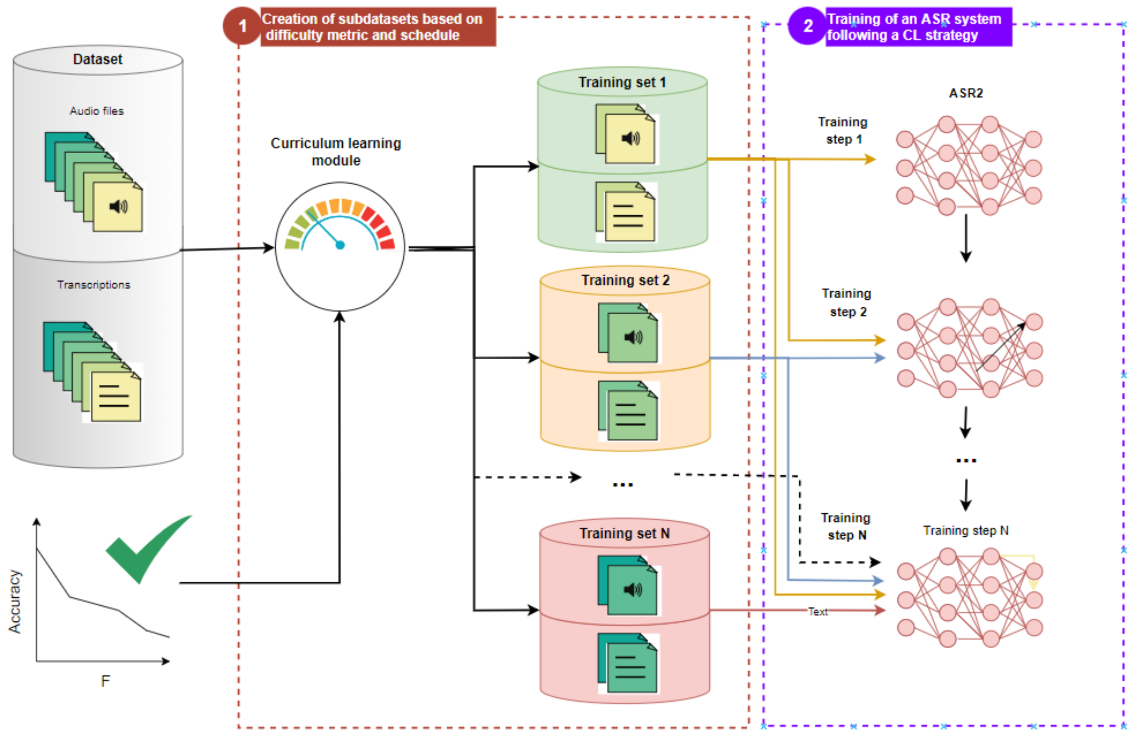

1 Creation of subdatasets based on difficulty metric and schedule
2 Training of an ASR system following a CL strategy
Dataset
Audio files
Transcriptions
Curriculum learning module
Training set 1
Training set 2
Training set N
...
Accuracy
F
ASR2
Training step 1
Training step 2
Training step N
Text
...

Figure 4: Training of an ASR system using a CL strategy based on one of the prosodic features candidates as difficulty metric.

3.2 Prosodic features

In the following section we introduce some characteristics of sound that are used in the field of prosody to describe speech and provide the list of prosodic features related to each characteristic that will be considered in our experiments[1].

[1] Descriptions provided here are based on:

- Ladefoged, P (1996). *Elements of Acoustic Phonetics*. Chicago: University of Chicago Press

3.2.1 Fundamental frequency, F0

In acoustic phonetics, the fundamental frequency, also called first harmonic or F0, refers to the lowest frequency component in the speech sound wave. It is based upon the number of complete cycles of vibration of the vocal folds.

Figure 5 shows the waveform produced when pronouncing Spanish vowel [a]. As it can be seen, the wave consists of an approximate repetition of the shaded pattern. The duration of the repeated pattern is 0.004 seconds, that is, in one second it is repeated 250 times. This is equivalent to say that the fundamental frequency is 250Hz.

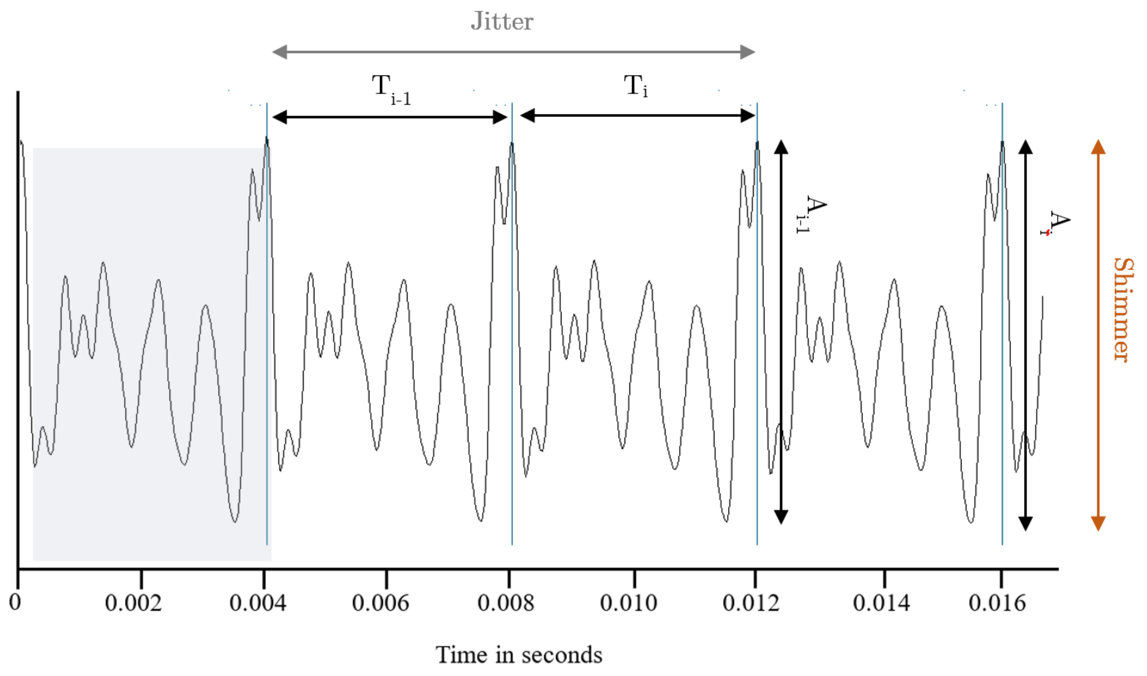


Figure 5. Waveform of a Spanish [a] pronounced by the author of this work. The shaded area shows a pattern that is repeated along the timeline; it corresponds to the lowest frequency component of the wave sound, F0. The horizontal double arrows show the intervals considered when computing Jitter. The vertical ones, the values of the amplitude considered when computing shimmer.

The fundamental frequency is of particular importance in studies of intonation, where it displays a close correspondence with the pitch, the auditory sensation of “melody”. Variations in pitch have phonological functions, as in word stress; syntactic functions, as

- Crystal, D. (2008). *A Dictionary of Linguistics and Phonetics.* Oxford: Blackwell.
- Teixeira, J.P., Oliveira, C., Lopes, C. (2013). *Vocal Acoustic Analysis – Jitter, Shimmer and HNR Parameters.* Procedia Technology, 9, 1112-1122
- Praat’s webpage: https://www.fon.hum.uva.nl/praat/

in the effect produced by a comma; and statement-level functions, to distinguish clause types (declarative, interrogative, imperative and exclamative).

The fundamental frequency at each point is computed considering a windows of a certain time length. Variations are then computed comparing the mean fundamental frequency consecutive windows. In our study, we have analysed the effect of the following parameters related to the fundamental frequency:

- **mean_f0**: Average value of the fundamental frequency along the utterance.
- **stdev_f0**: Standard deviation of the fundamental frequency along the utterance.
- **mean_delta_f0**: Average of the variation of the fundamental frequency along the sentence.
- **stdev_mean_f0**: Standard deviation of the variation of the fundamental frequency along the sentence.

3.2.2 Intensity

The intensity is the amplitude the signal of the audio sample. In speech, it corresponds to the size of the vibrations of the vocal folds causing the variations in air pressure that originate the sound wave. From prosodic perspective, intensity of signal is perceived as loudness.

In our study, we have analysed the effect of the following parameters related to the fundamental frequency:

- **mean_delta_intensity**: Average of the variation of the intensity along the sentence.
- **stdev_mean_intensity**: Standard deviation of the variation of the fundamental frequency along the sentence.

3.2.3 Harmonics-to-Noise Ratio (HNR)

The Harmonics-to-Noise Ratio (HNR) represents the degree periodicity. It is computed as:

$$HNR(dB) = 10\log\frac{E_{periodic}}{E_{noisy}}$$

where $E_{periodic}$ is the fraction of energy in the periodic part of the signal and E_{noisy} is the fraction of energy in the noisy part. In speech, the periodical component arises from the vibration of the vocal cords; the noisy one, from the glottal noise.

3.2.4 Jitter

Jitter is a measure of the deviation from true periodicity of a presumably periodic signal. It is defined as the period variation from cycle to cycle. When computed, a number N of cycles must be taken into account:

$$jitter(\text{seconds}) = \frac{\sum_{i=2}^{N}|T_i - T_{i-1}|}{N-1} \quad (1)$$

where T_i is the duration of the ith interval and N is the number of intervals.

In this work we have used the following metrics related to jitter:

- **local_absolute_jitter**: Average jitter over the utterance, i.e, N from Equation 1 is the total number of periods in the utterance.
- **local_jitter:** local_absolute_jitter, divided by the average period.
- **rap_jitter**: Relative Average Perturbation, which is a version of local jitter defined in terms of three consecutive intervals.
- **ppq5_jitter**: Five-point Period Perturbation Quotient (PPQ5), which is also a version of local jitter, this time defined in terms of five consecutive intervals.
- **ddp_jitter**: Relative mean absolute third-order difference of the point process (or, equivalently, the second-order difference of the interval process).

3.2.5 Shimmer

Shimmer is defined as the amplitude variation from cycle to cycle. It is computed as the average among a certain number of periods of the absolute base-10 logarithm of the difference between the amplitudes of consecutive periods, multiplied by 20:

$$Shimmer(\text{dB}) = \frac{1}{N-1}\sum_{i=1}^{N-1}\left|20\log\left(\frac{A_{i+1}}{A_i}\right)\right| \quad (2)$$

where A_i is the amplitude of the -ith interval and N is the number of intervals considered.

The metrics related to shimmer analysed in this work are:

- **localdb_shimmer:** Average shimmer over the utterance, i.e., N from Equation 2 is the total number of periods in the utterance.
- **local_shimmer:** Average absolute difference between the amplitudes of consecutive periods, divided by the average amplitude.
- **appq3_shimmer:** Three-point Amplitude Perturbation Quotient, the average absolute difference between the amplitude of a period and the average of the amplitudes of its direct neighbours, divided by the average amplitude.

- **appq5_ shimmer:** Five-point Amplitude Perturbation Quotient, the average absolute difference between the amplitude of a period and the average of the amplitudes of its four closest neighbours, divided by the average amplitude.
- **appq11_shimmer:** 11-point Amplitude Perturbation Quotient, the average absolute difference between the amplitude of a period and the average of the amplitudes its ten closest neighbours, divided by the average amplitude.

3.3 Evaluation

In both types of experiments, selection of candidate prosodic features and curriculum learning experiments, we will use the Word Error Rate (WER) to evaluate transcriptions. WER is a common metric in the field of ASR. It works at word level, providing a measure of how many "errors" are in the transcription text produced by system. It is computed as[2]:

$$WER = \frac{S + D + I}{N}$$

where *S* is the number of substitutions, *D*, the number of deletions, *I*, the number of insertions, and *N*, the number of words in the reference (gold transcription). Since it computes errors, rather than hits, a high WER indicates low performance.

In the selection of candidate prosodic features, we will use WER to score the level of difficulty of all utterances in the dataset collection: utterances for which ASR1 provides transcriptions with low WER will be considered as "easy". When reporting the performance in the CL training experiments, we will use WER accuracy instead, which is defined as:

$$W_{Acc} = 1 - WER$$

3.4 Experimental set-up

The following sections describe the experimental set-up: dataset collection containing speech utterances and gold transcriptions, speech analysis software used to extract prosodic features, ASR models used to compute accuracy and carry out experiments, and hardware.

[2] Font: Wikipedia contributors. (2020, February 7). Word error rate. In Wikipedia, The Free Encyclopedia. Retrieved 15:41, June 19, 2022, from https://en.wikipedia.org/w/index.php?title=Word_error_rate&oldid=939575741

3.4.1 The dataset collection: LibriSpeech

LibriSpeech (Panayotov et al., 2015) is a corpus of read speech, based on LibriVox's public domain audiobooks, built specifically to enable the training and testing of ASR systems. The collection contains 1000 hours of speech sampled at 16 kHz, organized in several subsets for training, evaluating, and testing. For each case, there are two kinds of subset: those labeled as “clean” and those labeled as “other”.

The collection has been built making a careful alignment between the recordings from LibriVox and the corresponding texts from the Project Guttenberg. The resulting pairs have been split into short fragments of 35 seconds or less to facilitate training. The selection of samples to be included in each subset has been made following these criteria:

- Subsets labeled with “clean” are, on average, of higher recording quality and with accents closer to US English.
- For all subsets there is gender balance in terms of number of speakers and available data.
- Subsets are speaker disjoint: all fragments corresponding to the same speaker are assigned to one and only one subset.

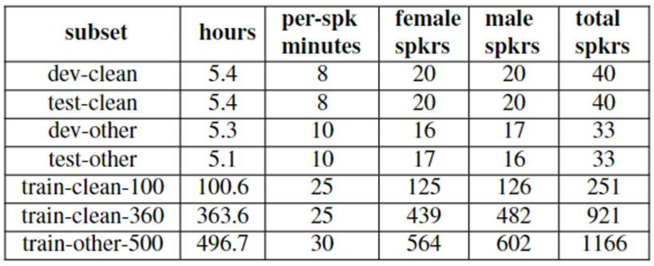
For each speaker in the clean training sets the total amount of speech is limited to 25 minutes. For validation and test sets, approximately eight minutes of speech is used.

The entire collection includes these subsets:

- **dev-clean**, **test-clean** - development and test set containing approximately 5 hours of "clean" speech.
- **train-clean-100** - training set, of approximately 100 hours of "clean" speech.
- **train-clean-360** - training set, of approximately 360 hours of "clean" speech.
- **dev-other**, **test-other** - development and test set containing approximately 5 hours of speech more "challenging" to recognize.
- **train-other-500** - training set of approximately 500 hours containing speech that was not classified as "clean".

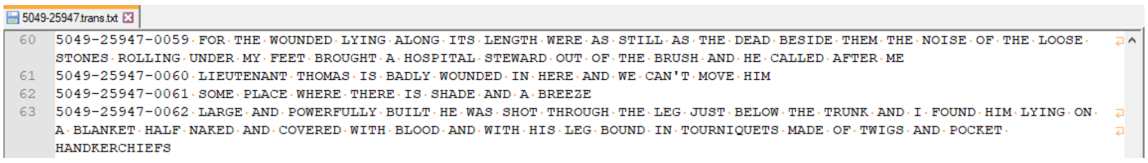
Table 1 includes detailed information of each subdataset.

Table 1. Table subsets in LibriSpeech



subset	hours	per-spk minutes	female spkrs	male spkrs	total spkrs
dev-clean	5.4	8	20	20	40
test-clean	5.4	8	20	20	40
dev-other	5.3	10	16	17	33
test-other	5.1	10	17	16	33
train-clean-100	100.6	25	125	126	251
train-clean-360	363.6	25	439	482	921
train-other-500	496.7	30	564	602	1166

Inside each subset, samples are organized in folders according to readers and chapters. For each reader and chapter, a text file with the transcription of the text fragments is included. Audio samples are provided in .flac (Free Lossless Audio Codec) format, an audio file format with lossless compression, and labelled with the ID of reader and the ID of the chapter in the LibriVox's database, plus a number indicating the fragment within the chapter. For instance, audio sample 5049-25947-0060.flac from train-clean-100 subset corresponds to reader 5049, chapter 25947 and fragment 60. Its gold transcription can be found in train-clean-100/5049/25947/5049-25947.trans.txt, shown in Figure 7. It reads: "5049-25947-0060 "lieutenant thomas is badly wounded in here and we can't move him". Text files with metadata for books, chapters and speakers is also provided.



```
5049-25947-0059 FOR THE WOUNDED LYING ALONG ITS LENGTH WERE AS STILL AS THE DEAD BESIDE THEM THE NOISE OF THE LOOSE
STONES ROLLING UNDER MY FEET BROUGHT A HOSPITAL STEWARD OUT OF THE BRUSH AND HE CALLED AFTER ME
5049-25947-0060 LIEUTENANT THOMAS IS BADLY WOUNDED IN HERE AND WE CAN'T MOVE HIM
5049-25947-0061 SOME PLACE WHERE THERE IS SHADE AND A BREEZE
5049-25947-0062 LARGE AND POWERFULLY BUILT HE WAS SHOT THROUGH THE LEG JUST BELOW THE TRUNK AND I FOUND HIM LYING ON
A BLANKET HALF NAKED AND COVERED WITH BLOOD AND WITH HIS LEG BOUND IN TOURNIQUETS MADE OF TWIGS AND POCKET
HANDKERCHIEFS
```

Figure 6. Extract from file train-clean-100/5049/25947/5049-25947.trans.txt

In this project, we have used the train-clean-100 subset for training and the dev-clean subset for validation. Train-clean-100 contains 28,539 samples; dev-clean, 2,703.

3.4.2 Prosodic features extraction: Praat and Parselmouth

Prosodic features have been extracted using the dedicated software described in the following sections.

3.4.2.1 Praat

Praat (Boersma & Weenink, 2022) is an open-source computer program for speech analysis in phonetics. It can analyse, synthesize, and manipulate speech. Particularly, it provides a graphic interface and a set of tools to generate, edit and analyse waveforms, spectrograms, intensity contours and pitch tracks.

Let’s see some of what the package offers with a practical example from our dataset collection, the audio sample 5049-25947-0060.flac from the train-clean-100 subset of the LibriSpeech collection. The sample corresponds to a fragment (sentence) of the book *Notes of a War Correspondent* (Davis, 1910). The text read by the volunteer reader is “lieutenant thomas is badly wounded in here and we can't move him”.

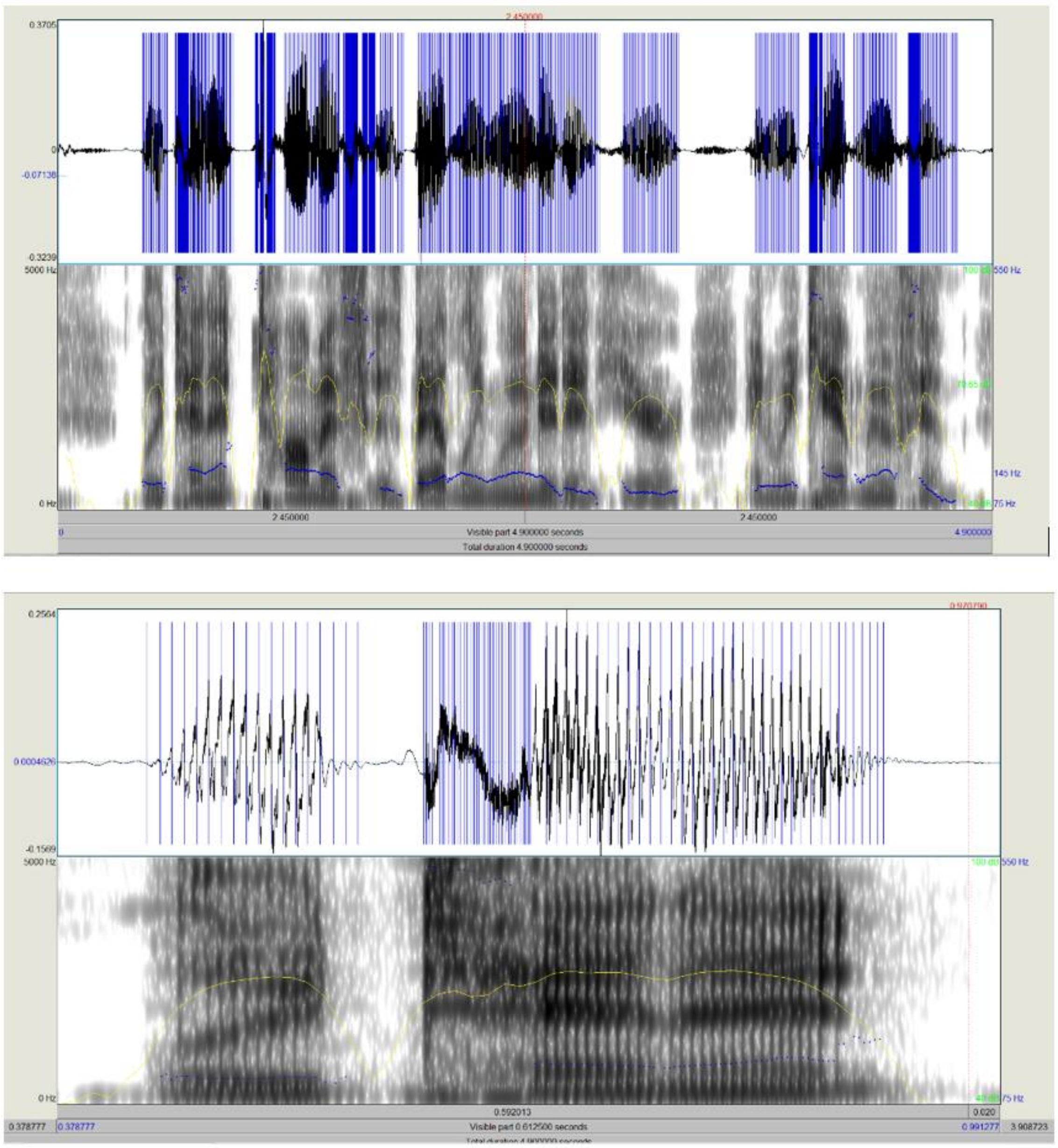


Figure 7. Audio sample 5049-25947-0060.flac as displayed by Praat.The screen above corresponds to sentence “lieutenant thomas is badly wounded in here and we can't move him”. In the screen below, the word “lieutenant” has been zoomed in.

Figure 7 shows some of the representations that the package can generate. All representations are plotted against the same horizontal axis: the timeline of the audio file. These representations are:

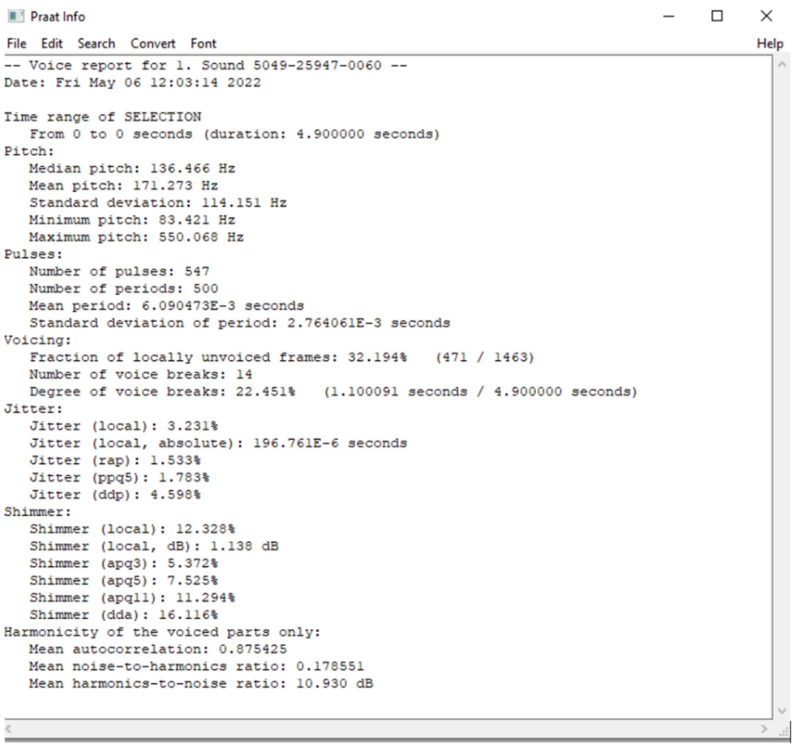
- **Waveform**: The waveform is the plot of the signal stored in the FLAC file after decompression. Sound is a pressure wave; this signal reproduces the signal picked up by the microphone when the audio was recorded.
- **Spectrogram**: The spectrogram is a representation of the energy density as a function of time (horizontal axis) and frequency (vertical axis). Darker parts of the spectrogram mean higher energy densities; lighter parts mean lower energy densities.
- **Pulses**: Pulses represent periodic fragments of the waveform. They are plotted as vertical blue straight lines on the waveform plot, at the maximum of each fragment. Pulses are the base for further computations such as the fundamental frequency, jitter or shimmer. They only exist if the waveform shows periodicity; otherwise, the waveform is considered as noise and no further computations can be done.
- **Pitch**: Pitch, or fundamental frequency, is shown as blue dots on the spectrogram area. For each pulse, pitch is computed by counting the number of pulses within a time window on both sides of the pulse mark, and dividing into the window length.
- **Intensity**: The intensity, plotted as a yellow line on the spectrogram area, represents the mean amplitude at linearly spaced time intervals.

With all the previous information, Praat can compute global parameters such as the ones used in this work. Figure 8 shows, for instance, the Voice report for the sample audio 5049-25947-0060.

3.4.2.2 Parselmouth

Parsemouth (Boersma & Weenink, 2021) is an open-source Python library that facilitates access to core functionality of Praat in Python, allowing researchers to integrate sophisticated acoustic analyses in their Python scripts. It uses the pybind11 library to communicate with and access to Praat's internal objects, memory, and code. The package includes Praat as part of the library in order to provide immediate access to the raw data calculated by Praat. This, together with the use of Numpy, makes it possible to use the existing data without copying, which makes the computation faster.

Among other functionalities, Parselmouth can quickly code up batch operations over a collection of files. This allows, for instance, to get a set of parameters for all audio files in a dataset collection in one run.



```
-- Voice report for 1. Sound 5049-25947-0060 --
Date: Fri May 06 12:03:14 2022

Time range of SELECTION
   From 0 to 0 seconds (duration: 4.900000 seconds)
Pitch:
   Median pitch: 136.466 Hz
   Mean pitch: 171.273 Hz
   Standard deviation: 114.151 Hz
   Minimum pitch: 83.421 Hz
   Maximum pitch: 550.068 Hz
Pulses:
   Number of pulses: 547
   Number of periods: 500
   Mean period: 6.090473E-3 seconds
   Standard deviation of period: 2.764061E-3 seconds
Voicing:
   Fraction of locally unvoiced frames: 32.194%   (471 / 1463)
   Number of voice breaks: 14
   Degree of voice breaks: 22.451%   (1.100091 seconds / 4.900000 seconds)
Jitter:
   Jitter (local): 3.231%
   Jitter (local, absolute): 196.761E-6 seconds
   Jitter (rap): 1.533%
   Jitter (ppq5): 1.783%
   Jitter (ddp): 4.598%
Shimmer:
   Shimmer (local): 12.328%
   Shimmer (local, dB): 1.138 dB
   Shimmer (apq3): 5.372%
   Shimmer (apq5): 7.525%
   Shimmer (apq11): 11.294%
   Shimmer (dda): 16.116%
Harmonicity of the voiced parts only:
   Mean autocorrelation: 0.875425
   Mean noise-to-harmonics ratio: 0.178551
   Mean harmonics-to-noise ratio: 10.930 dB
```

Figure 8. Voice report for the sample audio 5049-25947-0060

3.4.3 ASR systems

In this work, two ASR models have been used. In the experiments aimed at finding candidates for a CL strategy, ASR1, a pre-trained model, has been used to evaluate the difficulty of utterances from the training dataset. In the curriculum learning experiments, ASR2, a new model, has been trained from scratch using a CL strategy.

3.4.3.1 AR1: Hugging Face / Speechbrain

The ASR system used to assign a difficulty score to the samples in the training set has been the repository speechbrain/asr-wav2vec2-commonvoice-en from HuggingFace Transformers. Hugging Face Transformers (Wolf et al., 2020) is a toolkit designed to easily develop state-of-the-art speech technologies, including systems for speech recognition. The repository speechbrain/asr-wav2vec2-commonvoice-en provides all the necessary tools to perform automatic speech recognition from an end-to-end system pre-trained on Common Voice (English Language) (Ardila et al., 2020) within SpeechBrain (Ravanelli et al., 2021). In the current work, we will refer to this system as ASR1.

3.4.3.2 ASR2: Fairseq S2T

The ASR system used to perform experiments on CL strategies has been built upon Fairseq framework [Ott 2019], a state-of-the art, open-source sequence modelling toolkit developed by Facebook AI. It is based on Pytorch and it allows to train custom neural sequence end-to-end models for several text generation tasks such as machine translation, summarization and language modelling. The library provides command line tools to pre-process data, train and evaluate models, and generate output, together with a set of parameters to define advanced training options and select tasks, models, criterions (loss functions), optimizers and learning rate schedulers. The easiness and flexibility through which all these options can be set up make this framework a very suitable one for both research and production.

The Fairseq task that can be used as an ASR system is the speech-to-text task (S2T) [Wang 2020]. This task has been designed for both automatic speech recognition and speech-to-text translation.

In this report, we will refer to this system as ASR System 2. Since results concerning CL experiments are highly dependent on training parameters other than those strictly related to CL, we provide relevant information on the system configuration, so that results can be contextualized and reproduced.

Data pre-processing

Physically, sound is a pressure wave. Recording a sound means converting this physical waveform into an electrical representation by means of a microphone and storing it in some medium. In the case of digital audio, the continuous electrical signal must be first discretized (sampling) and the resulting values encoded into digital values (bits) of some precision (bit-depth). The FLAC audio files from LibriSpeech are sampled at 16kHz (1600 samples per second) and 16 bit-depth.

Fairseq library includes tools to download and pre-process several ASR datasets benchmarks. In our project, the LibriSpeech collection was downloaded and pre-processed using the `prep_librispeech_data.py` method. The process followed these steps:

1. **Downloading**. Datasets for training, testing and evaluation were downloaded. The original files are FLAC audio files, organised in folders according to the speaker id.

2. **Spectrogram generation**. As most ASR state-of-the-art systems, the Fairseq S2T task takes log-mel spectrograms as input. In this part of the process, Kaldi-compliant (Povey et al., 2011) 80-channel log mel-filter bank features are extracted from FLAC audio files and stored into numpy files via torchaudio.

3. **Zipping of features**: Filter banks are packed into a ZIP file for faster reading.
4. **Manifest generation**: A tabular separated values file (TSV) is generated with the following information for each sample: the ID, the path to the filter bank, the number of frames, the target text and the speaker ID.
5. **Vocabulary generation**: A vocabulary file with 10,000 unigrams is created via SentencePiece (Kudo & Richardson, 2018) tokenizer.
6. **Configuration file generation**: A YAML file is generated with online speech data transforms and data-related settings, such as tokenizer type and vocabulary path.

For the text data, Fairseq S2T makes online tokenization with the tokenizer defined in the yaml file.

Model

The model used to run our experiments was a vanilla transformer (Vaswani et al., 2017) included in the Fairseq library, the s2t_transformer. This version includes a convolutional downsampler before the positional embedding layer which reduces the length of speech inputs by 3/4th before they are fed into the encoder. In order to perform our experiments faster, the small configuration version of the model, s2t_transformer_s, was chosen. This version has about 30M parameters and is 350MBs in size.

Optimization

The loss function used to optimise the model parameters was cross-entropy. Adam optimizer was used, with a learning rate decreasing with an inverse square root scheduler. Before curriculum learning experiments, some fine-tuning for learning rate and weight decay was done to set the baseline, as well as tests including and not including a warm-up stage. The inclusion of a warm-up stage when training a vanilla Transformer from scratch using any gradient-based optimization approach is crucial for the final performance of the model [Xiong 2020]. These is because the vanilla Transformer places the layer normalisation between the residual blocks, resulting in large-expected gradients of the parameters near the output layer, which makes the training unstable. With the warm-up stage, the optimization starts with an extremely small learning rate and gradually increases up to a predefined maximum value in a predefined number of iterations.

Batching

To create batches, Faiseq groups source and target sequences of similar length. This minimises the sequences padding. The content batches stay the same throughout training;

batches themselves are shuffled randomly every epoch. The library also allows to accumulate gradients from multiple mini-batches before updating, resulting in larger effective batch sizes.

Mixed precision

Fairseq provides support for both full precision (FP32) and half precision (FP16). When setting the training to FP16, forward-backward computations are done in half precision, while parameters remain in full precision. This technique accelerates the process and preserves accuracy. A dynamic loss scale is applied to avoid underflows for activations and gradients.

3.4.4 Curriculum learning module: Speacher

To feed the system using CL strategies a dedicated module, Speacher (Cambara, 2021), has been used. Speacher (from "speech teacher") takes dataset manifests files as input and generates an output folder with subdatasets sampled according to difficulty criteria. The tool has been designed to work with other speech recognition frameworks, like Fairseq.

3.4.5 The hardware

As part of the project, systems ASR1 and ASR2, along with their dependencies, were installed in two servers at University of Barcelona and Universidad Nacional a Distancia. Especially CL experiments, which consisted in training a state-of-the-art ASR model with 30M parameters for a considerable number of epochs, required the use of powerful GPUs. Table 2 presents basic information of both servers.

Table 2. Short datasheet of servers used in the project.

	Server 1	Server 2
CPU	Xeon W-2155@3.3GHz (10 cores)	i9-10920X@3.50GHz (12 cores9
GPU	4x Nvidia RTX2080Ti of 12GB	2 Nvidia RTX 3090 of 24 GB
RAM	64Gb	128GB
Cuda version	10.1	11.5

4 Experiments and results

In this section we present the two types of experiments conducted as mentioned above. First, a collection of utterances from a dataset was analysed. Several prosodic features were extracted from each utterance using a dedicated software, Parselmouth. In parallel, a transcription for each utterance was generated and evaluated using ASR1, a pre-trained ASR model. As a result of this evaluation, an accuracy score in terms of WER was provided for each utterance in the dataset. Then, for each prosodic feature, we looked for relationships with the accuracy achieved. Those prosodic features showing a relationship with WER became candidates for the second kind of experiments. These latter experiments consisted in training a new ASR model, ASR2, from scratch using different strategies, in order to test whether the application of a curriculum based on one of the selected prosodic features improved the final accuracy and/or the training speed. In what follows we describe these experiments, as well as the results.

4.1 Selection of candidate prosodic features

This part of the work includes the extraction of prosodic features, the evaluation of transcriptions by ASR1, and the comparison between both results to find relationships.

4.1.1 Extraction of prosodic features

Prosodic features described in section 3.2 were extracted for each of the 28,539 utterances in the dataset collection, LibriSpeech train-clean-100, using the Parselmouth software. Table 3 shows statistical information of the results.

Table 3. Statistical information of prosodic features extracted from audio samples in dataset train-clean-100. Units are displayed in parentheses next-to the name of the feature.

Feature	**Minimum value**	**Maximum value**	**Mean value**	**Standard deviation**
mean_f0 (Hz)	59.10	317.00	162.00	45.20
stdev_f0 (Hz)	1.64	122.00	39.20	14.60
mean_delta_f0 (Hz)	-5.29	4.99	-0.11	0.31
stdev_delta_f0 (Hz)	0.26	22.50	6.13	2.58
mean_delta_intensity (dB)	-1.23	2.76	0.00	0.05
stdev_delta_intensity (dB)	0.13	18.20	1.86	0.98
hnr (dB)	-0.47	22.60	11.50	2.94
local_jitter	0.0098	0.0658	0.0239	0.0058
local_absolute_jitter (s)	0.00004	0.00076	0.00017	0.00008
rap_jitter (s)	0.002	0.031	0.010	0.003

ppq5_jitter (s)	0.003	0.038	0.011	0.003
ddp_jitter (s)	0.007	0.093	0.029	0.008
local_shimmer	0.023	0.226	0.098	0.021
localdb_shimmer (dB)	0.193	1.740	0.963	0.170
apq3_shimmer (dB)	0.009	0.120	0.036	0.011
aqpq5_shimmer (dB)	0.013	0.157	0.053	0.015
apq11_shimmer (dB)	0.025	0.561	0.098	0.026
dda_shimmer (dB)	0.026	0.360	0.107	0.034

4.1.2 Evaluation of the transcription quality by a pre-trained model

ASR1, described in section 3.4.3.1, was used to generate a transcription for each of the 28,539 utterances in the dataset collection, LibriSpeech train-clean-100. Each predicted transcription was evaluated by comparing it with its gold transcription. WER was used as the accuracy metric. Figure 9 shows the statistical information of the results. Each bin in the figure includes the number of utterances whose predicted transcriptions achieved the accuracy indicated in the horizontal label. As it can observed, one half of the samples in the dataset have WER lower than 10, and one third, lower than 20.

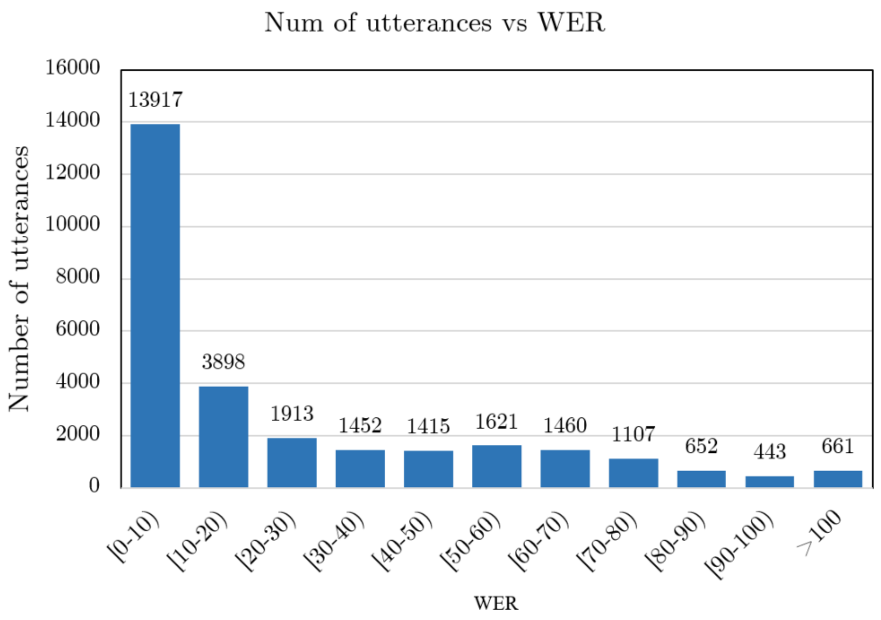
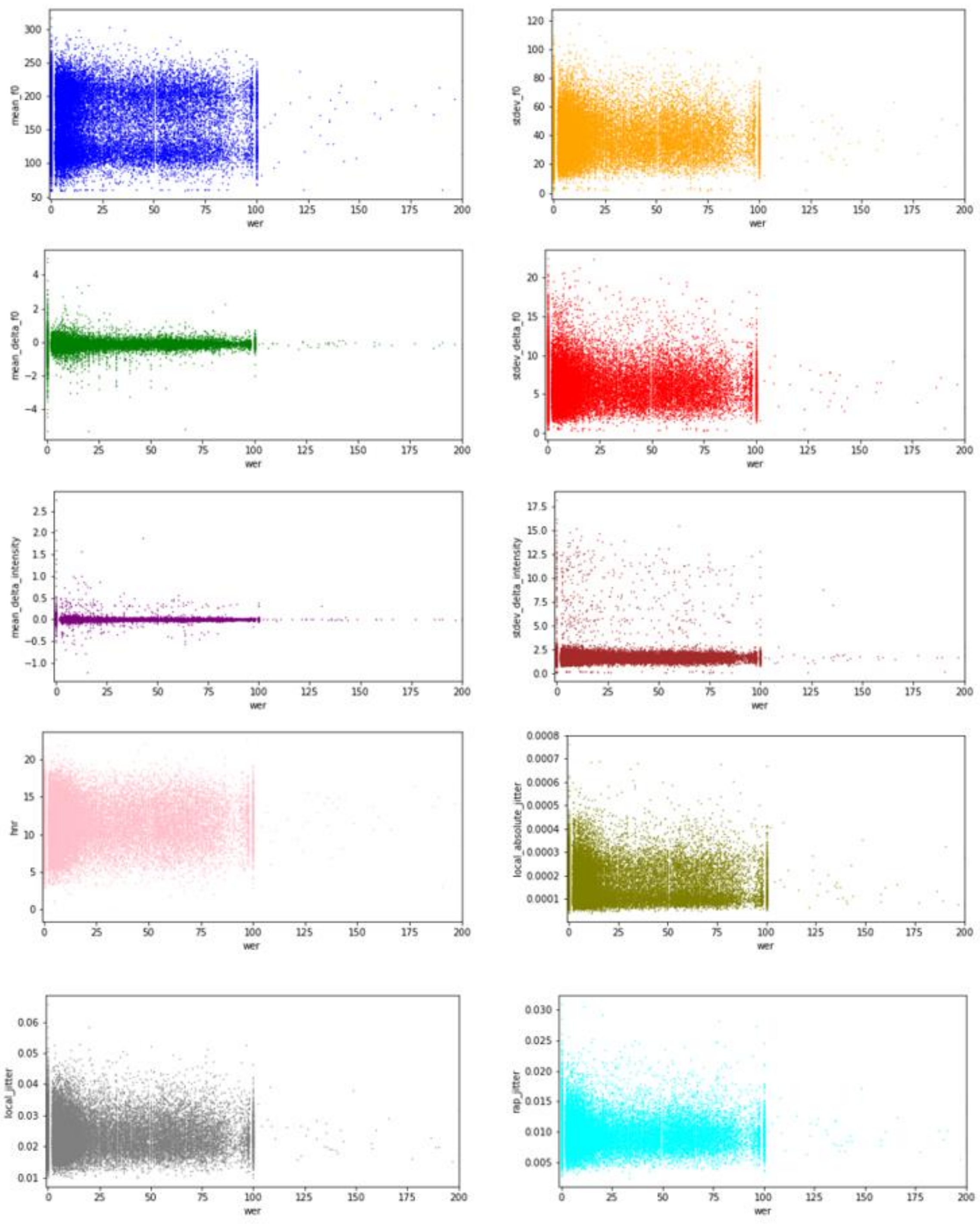


Figure 9. Statistical information of WER for transcriptions of utterances in dataset train-clean-100 as predicted by ASR1.

4.1.3 Selection of prosodic features for CL

Results from section 4.1.1 were compared with results from section 4.1.2 with the purpose of finding connections. We adopted two approaches: scatter graph and bin partition. In the scatter graph approach, 18 graphs were generated, one for each feature. The horizontal axis represented WER; the vertical axis, the prosodic feature being considered. For each utterance in the dataset, a dot was plotted in position (WER, prosodic feature value) for that utterance. Figure 10 shows the 18 resulting graphs.



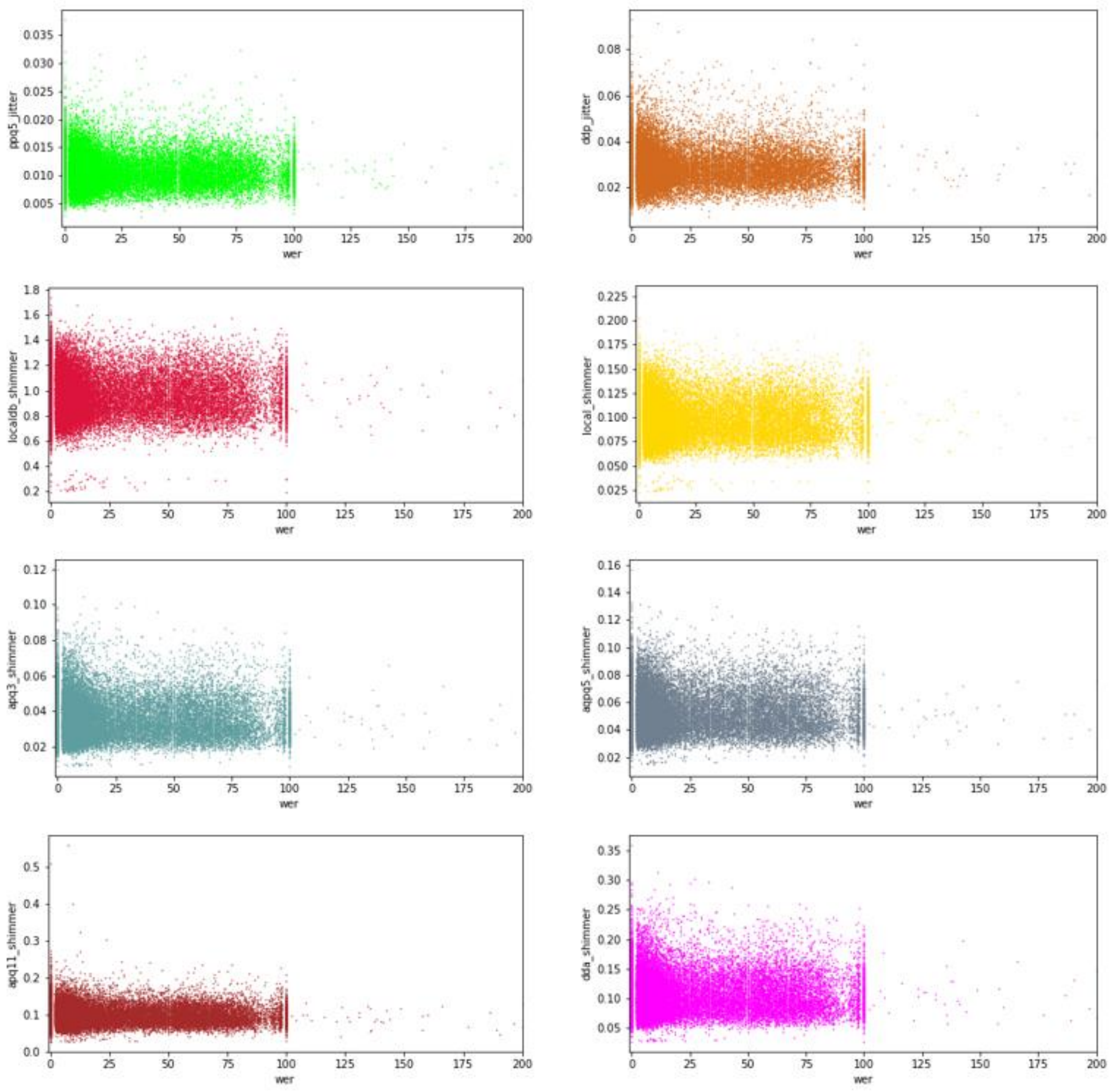


Figure 10. Prosodic features against WER for each sample in the train-clean-100 dataset. Each graph corresponds to the prosodic feature indicated in the vertical axis. Units are those showed in Table 3. Statistical information of prosodic features extracted from audio samples in dataset train-clean-100.

As it can be observed, all graphs show a quite homogeneous relationship between WER and the corresponding prosodic feature. Most dots are located along a horizontal band within the WER interval [0,100], which is centred very close to the mean value of the prosodic feature. This observation can be contrasted with the results of the linear regression computed on pairs (WER, prosodic feature), shown in Table 4. In all cases the slope is very close to zero. Its maximum value corresponds to stdev_f0 (0.00299), followed by mean_f0 (0.00296), stdev_delta_f0 (0.00025), mean_delta_f0 (0.00018) and HNR (0.00018). To perceive the centring of the band, we can compare values in column “mean value” in Table 3 with values in column “intercept” in Table 4: they are nearly the same. The Pearson correlation coefficient, which measures the strength of the linear relationship between two variables, is also close to zero, confirming the lack of direct linear relation.

Table 4. Linear regression between WER and each of the prosodic features considered.

Prosodic feature	**Slope**	**Intercept**	**Pearson correlation**
mean_f0	0.0029700	162.00	0.0049
stdev_f0	0.0029900	39.10	0.0153
mean_delta_f0	0.0000953	-0.112	0.0230
stdev_delta_f0	0.0002540	6.120	0.0074
mean_delta_intensity	-0.0000022	0.003	-0.0031
stdev_delta_intensity	-0.0001220	1.860	-0.0093
hnr	0.0001780	11.50	0.0045
local_jitter	-0.0000003	0.024	-0.0037
local_absolute_jitter	0.0000000	0.000	-0.0029
rap_jitter	0.0000003	0.010	0.0072
ppq5_jitter	0.0000001	0.011	0.0020
ddp_jitter	0.0000008	0.029	0.0072
local_shimmer	0.0000001	0.098	0.0004
localdb_shimmer	0.0000012	0.963	0.0005
apq3_shimmer	0.0000004	0.036	0.0027
aqpq5_shimmer	0.0000006	0.053	0.0030
apq11_shimmer	-0.0000006	0.098	-0.0016
dda_shimmer	0.0000012	0.107	0.0027

As for the width of the band, we can notice its tendency to be narrower with increasing WER. We believe, though, that this tendency does not show a relevant relationship between prosodic features and WER but should be attributed to the WER distribution among the dataset instead. The number of samples with low WER is considerably higher than the number of samples with high WER. As mentioned before, one half of the samples in the dataset have WER lower than 10, and one third, lower than 20. The concentration of dots at the left of the graph is much higher than at the right, which makes it more probable for feature values with high deviation from the mean to appear.

An observation we wish to make is the symmetry with respect to the zero value for graphs corresponding to mean_delta_f0 and mean_delta_intensity. These features represent variation among the sentence: a positive mean_delta_f0 indicates that the utterance ends in a higher pitch than it starts, and a negative, the reverse. Similarly, a positive mean_delta_intensity reveals that the volume at the end of the utterance is higher than at the beginning. A high absolute value of either of these parameters might be a sign of global intonation, regardless of if it is ascending or descending. In further experiments we took absolute values of these features into account, instead of the original ones. We will refer to the new features as abs(mean_delta_f0) and abs(mean_delta_intensity).

In the bin partition approach, we classified utterances into a set of bins according to the value of the prosodic feature being considered. Then, the average of WER for each bin was computed and plotted. For the sake of example, let's have a look to Figure 11, which corresponds to prosodic feature abs(mean_delta_f0). Six different partitions have been made, each with a different number of bins: N=1, 2, 3, 4, 5, 6. Each bin in a partition contains only those samples whose value for the prosodic feature being considered is within the interval displayed in the horizontal axis. Intervals have been computed so that each bin contains the same number of samples (quantile binning). In the first graph, where *N*=1, only one bin is considered, which includes 28,539 samples, the whole dataset. The range

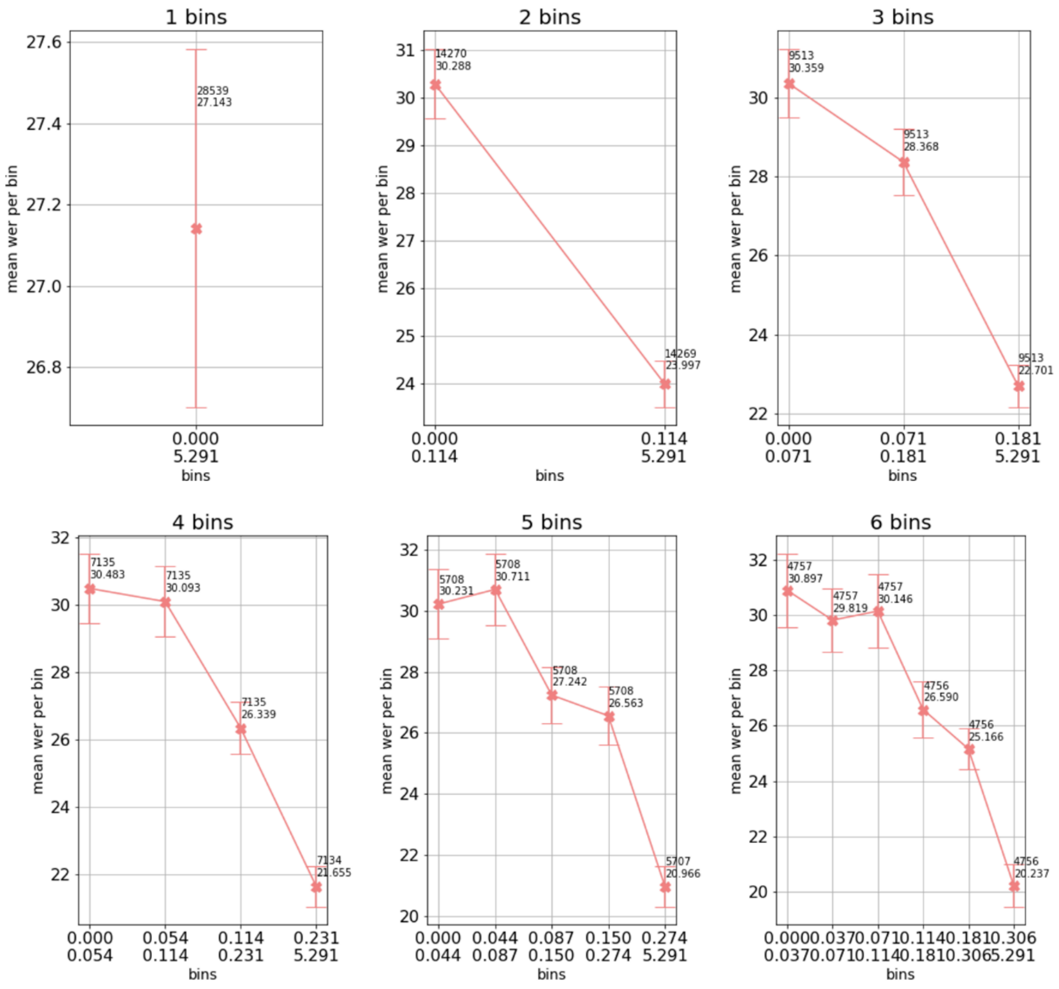


Figure 11. Average WER computed within quantiles of abs(mean_delta_f0)*. The horizontal axis displays intervals of* abs(mean_delta_f0)*; the vertical one, WER values. Dots displayed in the graph correspond to the mean WER computed over all utterances having a value of* abs(mean_delta_f0) *within the interval. For each dot, two annotations are included: first, the number of utterances included in the bin; and second, the value of the mean WER.*

of values of abs(mean_delta_f0) for utterances in this bin expands from the minimum value of this feature, 0, to its maximum, 5.291Hz. In the second graph, where *N*=2 (two bins), each bin contains half of the dataset, that is, 14,269 or 14,270 utterances. The range of values of the prosodic feature abs(mean_delta_f0) for the first and second bin are [0, 0.114) and [0.114, 5.291], respectively. This means that the 14,270 samples in the dataset have abs(mean_delta_f0)<0.114; the remaining 14,269 samples have abs(mean_delta_f0)≥0.114. The average WER for utterances having abs(mean_delta_f0)<0.114 is 30.288, whereas that for utterances having abs(mean_delta_f0) ≥0.114 is 23.997.

Using this approach, we do observe a relationship between feature abs(mean_delta_f0) and WER: an increase in the value of abs(mean_delta_f0) involves a decrease, in average, of WER. This is clearer for N≤4; for N>4, some deviations from this tendency appear. Qualitatively, this indicates that utterances that start and end with a significantly different pitch are, in average, easier to transcribe than utterances that start and end in a similar pitch.

An analogue partition was made using each of the 18 prosodic features described in Section 3.2. The resulting graphs can be consulted in Annex: Partitions of prosodic features. Apart from abs(mean_delta_f0), two more features showed an either decreasing or increasing relationship with WER: abs(mean_delta_intensity) and stdev_f0. Feature abs(mean_delta_intensity) showed a similar tendency as abs(mean_delta_f0): in average, a higher value of abs(mean_delta_intensity) resulted in a lower WER. That is, sentences starting in a lower volume that they end, or the other way round, are easier to transcribe than those starting and ending in similar values. By contrast, an increase of feature stdev_f0 resulted in an increase of WER.

We are not surprised by the results concerning abs(mean_delta_f0) and abs(mean_delta_intensity). They agree with results presented by (Goldwater et al., 2010), and they endorse our initial hypothesis that more modulated sentences might be easier to transcribe. However, results concerning stdev_f0 point to the opposite direction. This feature is a measure of how different is pitch along the utterance. For monotonous speech, pitch, that is, the fundamental frequency, is close to the mean all the time, so the standard deviation should be low. Conversely, highly modulated speech means that pitch deviate significantly from the mean at some moment, so the standard deviation should be high. The fact that utterances with low stdev_f0 are, in average, better transcribed by ASR1 seems to indicate that monotonous speech is easier to transcribe than more modulated speech.

4.2 Experiments using curriculum learning

In this part of the work, we trained a new ASR model, ASR2, from scratch using a CL strategy based on the previous results. We saw earlier that, when classified into bins according to the values of abs(mean_delta_f0), abs(mean_delta_intensity) and stdev_f0, a relationship with WER could be established. In particular, the average WER for subsets of utterances having lower values of abs(mean_delta_f0) or abs(mean_delta_intensity) was higher than that for subsets having higher values of abs(mean_delta_f0) or abs(mean_delta_intensity), respectively. By contrast, subsets of utterances having lower values of stdev_f0 had an average WER lower than that of subsets having higher values of stdev_f0. Following the CL hypothesis raised at the beginning of this work, these findings suggest that i) abs(mean_delta_f0), abs(mean_delta_intensity) and stdev_f0 might be good candidates for CL strategies and ii) an appropriate schedule might be to feed the system in N<=4 steps, adding a subdataset with more difficult samples each time.

Results concerning abs(mean_delta_f0) shows a parallelism with results reviewed by (Esteve-Gibert & Prieto, 2018). According to the authors, there is evidence that the direction of the intonation contour is used by pre-lexical infants to signal speech act information and may play a role in language acquisition. As described above, abs(mean_delta_f0) indicates the difference in pitch between the beginning and end of the utterance, which can be defined, precisely, as intonation contour. This provided us additional grounds to select this specific feature for our CL experiments. We believe that similar reasoning could have been applied to abs(mean_delta_intensity). Note, also, that the fact that the tendency shown by stdev_f0 contradicts what we would have expected considering human language comprehension does not mean that it should be rejected as a candidate for CL strategies. For any reason we do not know, it was easier for ASR1 to transcribe sentences having high stdev_f0. Consequently, the CL hypothesis could have been applied using this feature as a metric score as well.

Having all the above in mind, we undertook CL experiments using abs(mean_delta_f0) as our difficulty metric and a partition method with N=4 as our schedule. Consequently, the dataset was divided into 4 equal-sized subdatasets, each containing utterances whose value of abs(mean_delta_f0) was included in the intervals shown in the graph in Figure 11 corresponding to 4 bins. Table 5Table 3 lists the resulting datasets. The training of the system was made in 4 steps. In the first step, Dataset 1, which contained utterances with the higher value of abs(mean_delta_f0), was used to train the system for a step-length of L updates. In the second step, Dataset 2 was added, and training continued for another L updates. In the third step, Dataset 3 was added, and training continued for another L updates. Finally, in the fourth step, Dataset 4 was added, and training continued until a total number of 300,000 steps or stabilization.

Table 5. Subdatasets used in the CL strategy.

Label	Number of utterances	Values of **abs(mean_delta_f0)** (Hz)
Dataset 1	7135	[0.000, 0.054)
Dataset 2	7135	[0.054, 0.114)
Dataset 3	7135	[0.114, 0.231)
Dataset 4	7134	[0.231, 5.291]

The strategy with $N=1$, that is, the inclusion of the whole dataset from the beginning of the training, constituted our baseline. We also trained the system using a null strategy, which consisted in making a partition with N=4, like that of the CL strategy, but including random samples in each bin. The anti-curriculum approach, that is, training the system by feeding it with difficult samples at the beginning was also made. We undertook two types of training: with and without a warm-up stage. A warm-up stage is a period at the beginning of the training were the learning rate increases up to a certain value, before starting to decrease in the usual way. As we will see, the inclusion of a warm-up stage when training a vanilla Transformer from scratch using gradient-based optimization approach is crucial for the final performance of the model. However, it includes some complexity to our experiments, since the warm-up period overlaps with the curriculum learning one.

Altogether, our experiments can be summarised as follows:

1. Fine-tuning of two baselines, without and with warm-up.
2. Testing of null hypothesis, without and with warm-up.
3. CL and anti-CL experiments using a 4 bin-schedule based on abs(mean_delta_f0), without and with warm-up.

For the interpretation of results, we must mention that they are based on isolated experiments, i.e., no statistical approaches have been taken. Our purpose in this part of the project was merely to test the ground for further experiments.

4.2.1 Baseline

Before starting our CL experiments, we defined baselines based on the same ASR system and training dataset. We fined-tuned a baseline without warm-up, and one with warm-up. Two parameters were optimized: learning-rate and weight-decay. The values achieving best performance were used in all further experiments.

We began by fine-tuning the learning rate for the no-warm-up case. The precision was set to FP16 to speed up the process. The weight-decay was fixed to 0. Figure 12 shows accuracy evolution when training with learning rates within 0.0008 and 0.002. Learning rates from 0.01 resulted in gradient explosion and are not shown in figure. The best accuracy was obtained for *lr*=0.002.

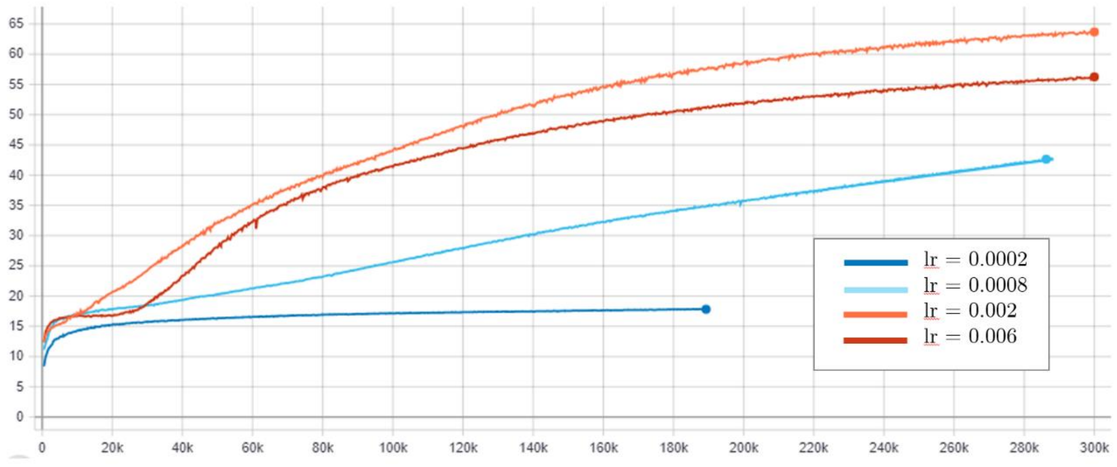


Figure 12. Fine-tuning of the learning rate for a baseline without warm-up.

The next parameter we fine-tuned was weight-decay. Figure 13 shows accuracy evolution when training in the same conditions as above, now with a fixed learning rate (0.002) and changing the weight-decay. Setting the weight-decay to 0.2 improved training from a final accuracy of 63.68 to 66.18 on the evaluation dataset, dev-clean.

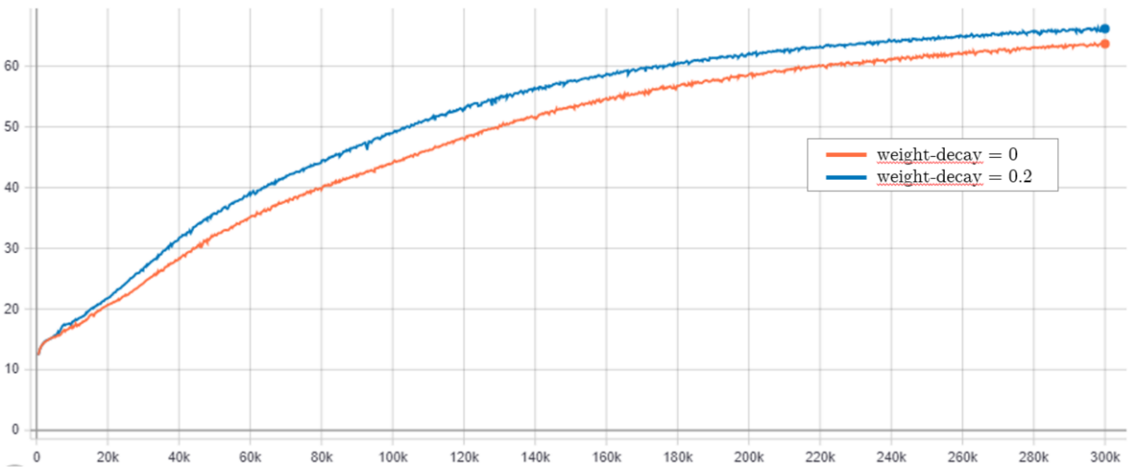


Figure 13. Fine-tuning of the weight-decay for a baseline without warm-up.

A baseline for trainings using warm-up was also fine-tuned. Here, we first fixed the learning-rate to 0.002 and changed the weight decay. Trainings with weight-decay equal or higher than 0.3 resulted in gradient explosion and are shown. Figure 14 shows accuracy evolutions for these experiments. As a result of these experiments, weight decay for the warm-up case was set to 0.2.

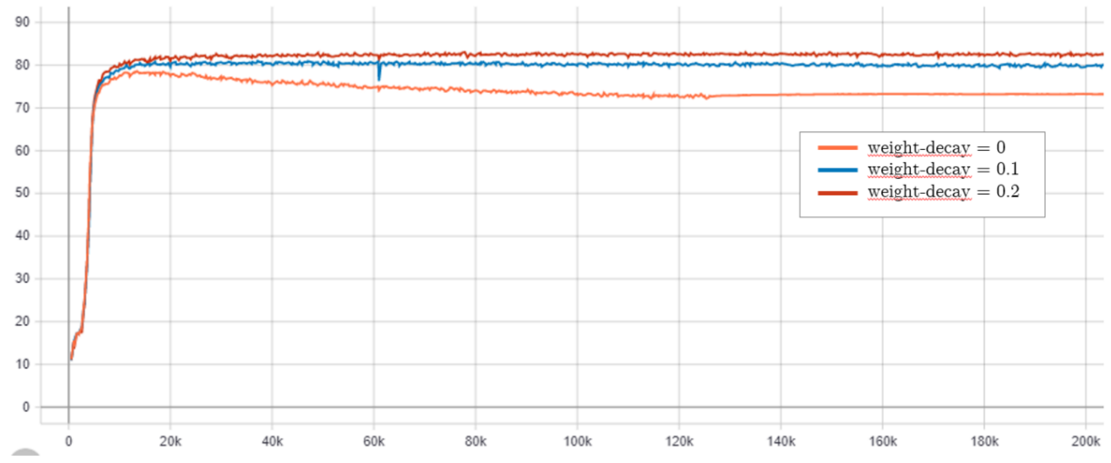


Figure 14. Fine-tuning of the weight-decay for a baseline with warm-up.

Finally, lower learning rates were tested for the warm-up case to see if accuracy case could be further improved. Figure 15 shows these experiments. As in the training without warm-up, the best learning rate was 0.002.

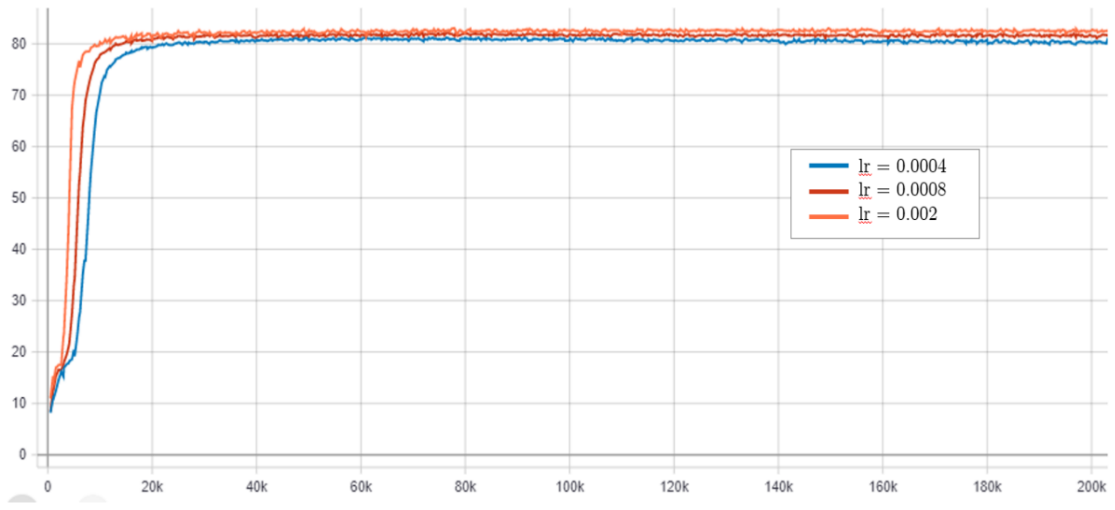


Figure 15. Fine-tuning of the learning rate for a baseline with warm-up.

Note the improvement achieved when using a warm-up stage. At 300,000 updates, the maximum allowed in our experiments, the best accuracy reached when training without warm-up is 64.0, whereas in the warm-up case, an accuracy of 82.3 is achieved in one tenth of that time, i.e at 30,000 updates.

4.2.2 Null hypothesis

The null hypothesis consisted in training the system following the same 4-step schedule as in the CL experiments. In this case, though, samples included in each subdataset were selected randomly. The purpose of this experiment was to compare the effects of sorting the data based in a difficulty metric and the simple effect of feeding the system in 4 steps. The step-length, that is, the number L of updates of the first 3 steps of the curriculum schedule, was set 10,000 for experiments without warm-up and 1,000 for experiments with warm-up. As training without warm-up is so slow, we considered that a large step-length was necessary to make the effects of curriculum learning evident. By contrast, in the warm-up case, it was important to fit the schedule within the warm-up stage, so that effects would not be mixed.

Figure 16 and Figure 17 show the evolution of the accuracy, evaluated on dev-clean dataset. The effect of adding new samples in the null-hypothesis case can be clearly seen in the no-warm-up case, were step-lengths are longer. Note that, in both the no-warm-up and warm-up cases, accuracy when training with the null hypothesis is behind the baseline, that is, feeding the system in 4 steps, adding a quarter of the data at each step, decreases training efficiency. In the warm-up case, training using the null-hypothesis speeds up between updates 6.000 and 15.000 reaching the baseline, but in the no-warm-up case there are residual effects of this delay still in update 300,000. It does not surprise us that training evolution goes behind at the beginning, when the number of samples fed into the system is smaller. Recall that, in the null hypothesis, samples in each bin are selected randomly and are not expected to be especially informative for the system. Overall, during the first steps the system has less information to fit the model. What should concern us is that, in the warm-up case, this effect is still significant after 300,000 updates.

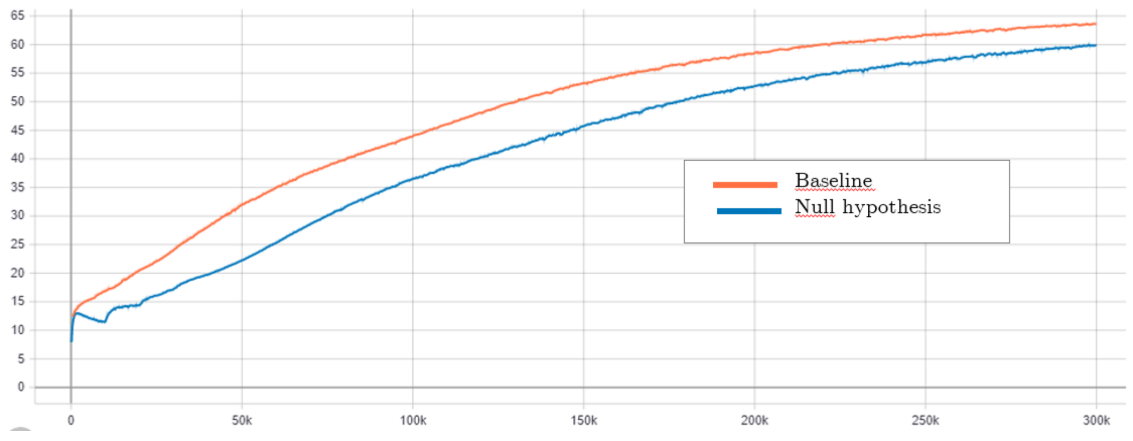


Figure 16. Comparison of accuracy evolution for baseline and null hypothesis, without warm-up.

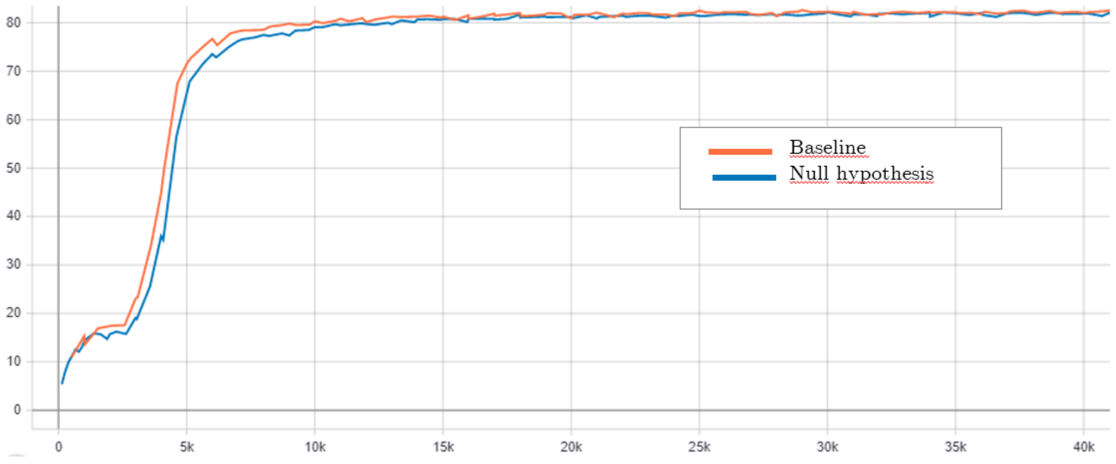


Figure 17. Comparison of accuracy evolution for baseline and null hypothesis, with warm-up.

4.2.3 CL experiments using abs(mean_delta_f0)

In this part of the work, we tested a CL design using a 4-bin schedule based on the prosodic feature abs(mean_delta_f0). Two schedules were tested. The first one, the "CL schedule", consisted in feeding the system with subdatasets from Table 5 in descending order of abs(mean_delta_f0), that is, from "easier" samples to "more difficult" ones. In the second schedule, the "Anti-CL schedule", the reverse was followed: datasets were included into the system in ascending order of abs(mean_delta_f0). Each pair of experiments were conducted twice: without and with warm-up. In experiments without warm-up the step length was fixed to 10,000; in those with warm-up, to 1,000. Figure 18 and Figure 19 show the evolution of the accuracy, evaluated on dev-clean dataset. In both figures, the null hypothesis has been added, to allow comparison.

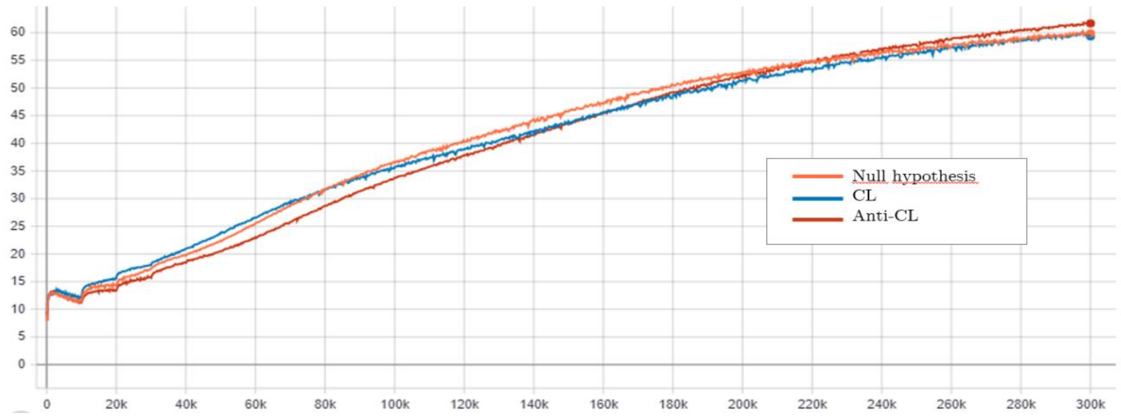


Figure 18. Comparison of accuracy evolution for CL and anti-CL strategies based on abs(mean_ delta_ f0), without warm-up. The null hypothesis is also included.

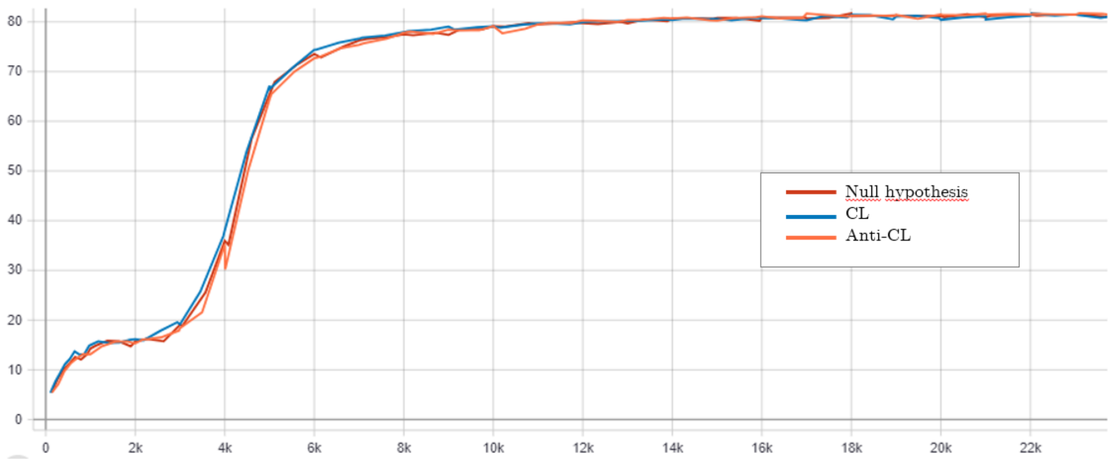


Figure 19. Comparison of accuracy evolution for CL and anti-CL strategies based on abs(mean_ delta_f0), with warm-up.The null hypothesis is also included.

Note that, during the first 70,000 and 8,000 updates in the no-warm-up and warm-up cases, respectively, the CL strategy outperforms the null-hypothesis, whereas the Anti-CL strategy underperforms it. Later, this effect is diluted. In the warm-up case, there is no significant difference in accuracies achieved in later epochs. In the no-warm-up case, the tendency even reverses, and the anti-CL strategy takes advantage. It should be studied if this effect is directly related to CL, or it is a random effect appearing at advanced stages, where the effects of the first epochs are somehow forgotten.

Table 6 and Table 7 summarize accuracy results for all experiments at updates 50,000 and 5,000 for the no-warm-up and warm-up cases, respectively. Results at the end of the training are also included. As we can see, in the no-warm-up case, at step 50,000, accuracy for the CL strategy is 5.9% above the null hypothesis and 9.7% above anti-CL; in the warm-up case, at step 5,000, accuracy for the CL strategy is 2.71% above the null hypothesis and 0.12% above anti-CL. Compared to the baseline, all experiments underperform during the first epochs. In the warm-up case, though, null, CL and anti-CL experiments reach the baseline at some point, so the final accuracy is quite similar. In the no-warm-up case, the accuracy of the baseline is better from the beginning to the end of the training, at 300,000 steps.

Table 6. Accuracies of experiments without warm-up

Experiment	**Accuracy at update 50,000**	**Accuracy at update 300,000**
Baseline	35.7	66.21
Null hypothesis	22.39	59.86
CL	23.71	59.34
Anti-CL	20.41	61.67

Table 7. Accuracies of experiments with warm-up

Experiment	**Accuracy at update 5,000**	**Accuracy at update 30,000**
Baseline	71.53	82.29
Null hypothesis	65.38	82.19
CL	67.15	81.46
Anti-CL	65.46	81.98

These results seem to indicate that the CL hypothesis based on abs(mean_delta_f0) has some positive effect in speeding up training during the first stages. However, this effect is not powerful enough as to compensate the negative effect that feeding the system in four steps produces. This effect can be isolated by comparing the baseline and the null hypothesis. For the no-warm-up case, accuracy when using the null hypothesis is 37% behind the baseline at update 50,000 and 9.6% at update 300,000. For the warm-up case, it is 8.6% behind at update 5,000 and 0.12% at update 30,000. Figure 16 and Figure 17 above clearly show this effect.

5 Conclusions and future work

In this work we have explored the possibility of improving performance of end-to-end ASR systems using CL strategies based on prosody. In the field of ASR, CL has been successfully applied using different approaches. This is the first attempt to take advantage of prosody effects.

In the first part of the project, several prosodic features have been extracted from a collection of 28,539 utterances. The same utterances have been transcribed using a pre-trained ASR system. The percentage of errors in these transcriptions has been used as difficulty metric and plotted against each prosodic feature using two different approaches: scatter graph and bin partition. In the scatter graph approach each utterance of the dataset was plot in position (x, y) of a 2-dimensional graph, where x was the number of errors, expressed as WER, and y was the value of the prosodic feature being considered. This approach did not allow us to detect any relationship between prosodic features and WER. In the bin partition approach, the collection of utterances was classified into a set of bins according to the value of the prosodic feature being considered. Then, the average of WER for each bin was computed and plotted. In this approach, WER showed a tendency to decrease with increasing values of features abs(mean_delta_f0) and abs(mean_delta_intensity), and a tendency to increase with increasing values of stdev_f0. The tendency of WER to either decrease or increase with ascending values of these features was clearer when the number of bins in the partition was equal or less than four. Feature abs(mean_delta_f0) corresponds to the absolute value of the mean variation of the fundamental frequency along the utterance. It is a measure of intonation contour: ultimately, it indicates the difference in pitch between the beginning and end of the utterance. Similarly, abs(mean_delta_intensity), or the absolute value of the mean variation of the intensity along the utterance, is a measure of volume contour. Feature stdev_f0 corresponds to the standard deviation of the fundamental frequency along the utterance.

In the second part of the project, a CL strategy was defined using feature abs(mean_delta_f0) as the difficulty metric. The original dataset of 28,539 was divided into 4 subdatasets, each containing a quarter of the samples. The partition was made so that in each subdataset there were utterances with value of abs(mean_delta_f0) within a certain interval. Then, a new ASR system was trained from scratch using this CL schedule: first, the system was trained using the dataset containing samples with the highest values of abs(mean_delta_f0) ("easy utterances", for either 1,000 or 10,000 steps, depending on whether a warm-up stage was included. Then, a new subdataset, the one containing utterances with the highest values of abs(mean_delta_f0) among the three remaining datasets, was added. Training continued for another 1,000 or 10,000 updates. Then the next dataset was added, and so on, until the whole collection was introduced. Experiments using the reverse strategy, i.e., training in 4 steps beginning with the dataset containing utterances with the lowest values of abs(mean_delta_f0) ("difficult utterances"), were also done. We refer to this strategy as anti-CL. A strategy using the same partition but including random samples in each subdataset was also tested. We refer to this strategy as the null hypothesis. All these experiments were done twice, with and without a warm-up stage. Results were compared with a baseline, which consisted in training the system using the whole dataset from the beginning.

For both the warm-up and no-warm-up experiments, CL strategies outperformed the null hypothesis and the anti-CL hypothesis at the beginning of the training. However, they did not outperform the baseline. This indicates that curriculum learning based on abs(mean_delta_f0) somehow helps the system to learn during the first stages, but this effect is not powerful enough as to compensate the negative consequences that feeding the system starting with a fraction of the dataset has.

Clearly, further experiments should be made before confirming or refuting the CL hypothesis based on abs(mean_delta_f0). The effectiveness of CL is highly sensitive to the mode of progression through the tasks (Graves et al., 2017). The CL hypothesis is grounded on the idea that training with "easy" examples avoids wasting time with noisy or harder to predict samples and helps the system achieve a better generalization (Bengio et al., 2009) faster. However, "easy" examples might carry too little information for the system to learn, delaying learning. The challenge behind the definition of a CL strategy is to find a balance between these two effects, both by selecting the appropriate difficulty metric and determining an efficient schedule. Apart from this, applying CL implies starting training with a reduced dataset, which has proved to have a negative effect. To reduce this effect, it might be more appropriate to use a 2-bin schedule, instead of a 4-bin one. Such a schedule would also magnify the difference between samples in the datasets, making the effect of CL more evident. The use of a larger dataset could also help to prevent the learning delay at the beginning, caused to a lack examples. Other step lengths, that is, the number of updates during which the system is trained before more data is added, should also tested, as well as other types of schedules, such as a continuous linear or exponential pacing.

On the other hand, alternative features could also be used to define new prosody-based CL strategies. Experiments in the first part of the project showed that abs(mean_delta_intensity) and stdev_f0 could be feasible candidates. A combination of abs(mean_delta_f0), abs(mean_delta_intensity) and stdev_f0 could also be used. In fact, a plot in the partition bin approach using the addition of normalised abs(mean_delta_f0) abs(mean_delta_intensity) unveiled an more pronounced relationship with WER.

Finally, different prosodic features should be analysed. Two features that we intend to explore in the future are mean(abs_delta_f0) and mean(abs_delta_intensity). Features considered in this work, abs(mean_delta_f0) and abs(mean_delta_intensity), are computed by first extracting local variations of an utterance, averaging over them, and finally taking the absolute value of this average. With this method, positive and negative variations within the utterance cancel out when computing the average, and only the difference between the beginning and the end remains. However, for speech comprehension, internal variations in intonation and volume are crucial, both at lexical (word stress) and syntactical level. Considering local absolute variations and averaging over those would provide a more significant scalar feature to give account of this important prosodic effect.

Bibliography

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., & others. (2016). Deep speech 2: End-to-end speech recognition in english and mandarin. *International Conference on Machine Learning*, 173–182.

Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F. M., & Weber, G. (2020). Common voice: A massively-multilingual speech corpus. *LREC 2020 - 12th International Conference on Language Resources and Evaluation, Conference Proceedings*, 4218–4222.

Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). Curriculum learning. *Proceedings of the 26th Annual International Conference on Machine Learning*, 41–48.

Boersma, P., & Weenink, D. (2022). *Praat: doing phonetics by computer* (6.2.12). http://www.praat.org/

Braun, S., Neil, D., & Liu, S.-C. (2017). A curriculum learning method for improved noise robustness in automatic speech recognition. *2017 25th European Signal Processing Conference (EUSIPCO)*, 548–552.

Cambara, G. (2021). *Speacher*. https://github.com/gcambara/speacher.

Chan, W., Jaitly, N., Le, Q., & Vinyals, O. (2016). Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2016-May*, 4960–4964. https://doi.org/10.1109/ICASSP.2016.7472621

Chang, X., Zhang, W., Qian, Y., le Roux, J., & Watanabe, S. (2019). MIMO-Speech: End-to-end multi-channel multi-speaker speech recognition. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 237–244.

Davis, R. H. (1910). *Notes of a war correspondent*. Library of Congress Online Catalog.

Elman, J. L. (1993). Learning and development in neural networks: the importance of starting small. *Cognition*, *48*(1), 71–99. https://doi.org/10.1016/0010-0277(93)90058-4

Esteve-Gibert, N., & Prieto, P. (2018). Early development of prosody-meaning interface. *The Development of Prosody in First Language Acquisition*, 228–246.

Farrús, M., Hernando, J., & Ejarque, P. (2007). Jitter and shimmer measurements for speaker recognition. *Proceedings of the Annual Conference of the International*

Speech Communication Association, INTERSPEECH, *2*, 1153–1156. https://doi.org/10.21437/INTERSPEECH.2007-147

Ghahremani, P., Babaali, B., Povey, D., Riedhammer, K., Trmal, J., & Khudanpur, S. (2014). A pitch extraction algorithm tuned for automatic speech recognition. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2494–2498. https://doi.org/10.1109/ICASSP.2014.6854049

Goldwater, S., Jurafsky, D., & Manning, C. D. (2010). Which words are hard to recognize? Prosodic, lexical, and disfluency factors that increase speech recognition error rates. *Speech Communication*, *52*(3), 181–200.

Graves, A. (2012). Sequence transduction with recurrent neural networks. *ArXiv Preprint ArXiv:1211.3711.*

Graves, A., Bellemare, M. G., Menick, J., Munos, R., & Kavukcuoglu, K. (2017). Automated curriculum learning for neural networks. *International Conference on Machine Learning*, 1311–1320.

Graves, A., Mohamed, A., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 6645–6649.

Hahn, L. D. (1999). *Native speakers' reactions to non-native stress in English discourse.* University of Illinois .

Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., & others. (2014). Deep speech: Scaling up end-to-end speech recognition. *ArXiv Preprint ArXiv:1412.5567.*

Isik, Y., Roux, J. le, Chen, Z., Watanabe, S., & Hershey, J. R. (2016). Single-Channel Multi-Speaker Separation Using Deep Clustering. *Proc. Interspeech 2016*, 545–549. https://doi.org/10.21437/Interspeech.2016-1176

Kano, T., Sakti, S., & Nakamura, S. (2017). Structured-Based Curriculum Learning for End-to-End English-Japanese Speech Translation. *Proc. Interspeech 2017*, 2630–2634. https://doi.org/10.21437/Interspeech.2017-944

Kim, S., Seltzer, M. L., Li, J., & Zhao, R. (2017). *Improved training for online end-to-end speech recognition systems.* http://arxiv.org/abs/1711.02212

Kim, S., Seltzer, M., Li, J., & Zhao, R. (2018). Improved Training for Online End-to-end Speech Recognition Systems. *Proc. Interspeech 2018*, 2913–2917. https://doi.org/10.21437/Interspeech.2018-2517

Kudo, T., & Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. *EMNLP 2018 - Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Proceedings*, 66–71. https://doi.org/10.18653/v1/d18-2012

Kuznetsova, A., Kumar, A., & Tyers, F. M. (2021). *A bandit approach to curriculum generation for automatic speech recognition.* http://arxiv.org/abs/2102.03662

Li, X., Tao, J., Johnson, M. T., Soltis, J., Savage, A., Leong, K. M., & Newman, J. D. (2007). Stress and emotion classification using jitter and shimmer features. *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, *4*, IV–1081.

Panayotov, V., Chen, G., Povey, D., & Khudanpur, S. (2015). Librispeech: an asr corpus based on public domain audio books. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5206–5210.

Platanios, E. A., Stretcu, O., Neubig, G., Póczos, B., & Mitchell, T. M. (2019). Competence-based Curriculum Learning for Neural Machine Translation. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 Conference of the North American* (pp. 1162–1172). Association for Computational Linguistics. https://doi.org/10.18653/v1/n19-1119

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., & Vesely, K. (2011). The Kaldi Speech Recognition Toolkit. *IEEE Signal Processing Society.*

Ranjan, S., & Hansen, J. H. L. (2021). Curriculum Learning based approaches for robust end-to-end far-field speech recognition. *Speech Communication*, *132*, 123–131.

Ravanelli, M., Parcollet, T., Plantinga, P., Rouhe, A., Cornell, S., Lugosch, L., Subakan, C., Dawalatabad, N., Heba, A., Zhong, J., Chou, J.-C., Yeh, S.-L., Fu, S.-W., Liao, C.-F., Rastorgueva, E., Grondin, F., Aris, W., Na, H., Gao, Y., ... Bengio, Y. (2021). *SpeechBrain: A General-Purpose Speech Toolkit.* http://arxiv.org/abs/2106.04624

Sainath, T. N., He, Y., Li, B., Narayanan, A., Pang, R., Bruguier, A., Chang, S., Li, W., Alvarez, R., Chen, Z., & others. (2020). A streaming on-device end-to-end model surpassing server-side conventional model quality and latency. *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6059–6063.

Salverda, A. P., Dahan, D., & McQueen, J. M. (2003). The role of prosodic boundaries in the resolution of lexical embedding in speech comprehension. *Cognition*, *90*(1), 51–89.

Slyh, R. E., Nelson, W. T., & Hansen, E. G. (2008). *Analysis of mrate, shimmer, jitter, and F/sub 0/ contour features across stress and speaking style in the SUSAS database*. 2091–2094 vol.4. https://doi.org/10.1109/ICASSP.1999.758345

Soderstrom, M., Seidl, A., Nelson, D. G. K., & Jusczyk, P. W. (2003). The prosodic bootstrapping of phrases: Evidence from prelinguistic infants. *Journal of Memory and Language*, *49*(2), 249–267.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*.

Vydana, H. K., Srivastava, B. M. L., Shrivastava, M., & Vuppala, A. K. (2016). Starting small learning strategies for speech recognition. *2016 IEEE Annual India Conference (INDICON)*, 1–6.

Wang, C., Tang, Y., Ma, X., Wu, A., Okhonko, D., & Pino, J. (2020). *fairseq S2T: Fast Speech-to-Text Modeling with fairseq*. http://arxiv.org/abs/2010.05171

Wang, C., Wu, Y., Liu, S., Zhou, M., & Yang, Z. (2020). *Curriculum Pre-training for End-to-End Speech Translation*. http://arxiv.org/abs/2004.10093

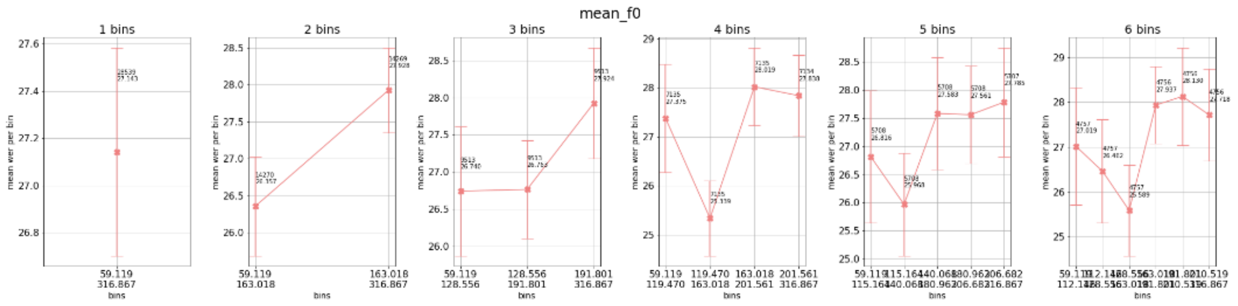
Wang, X., Chen, Y., & Zhu, W. (2020). A Survey on Curriculum Learning. *International Journal of Computer Vision*, *130*, 1526–1565. http://arxiv.org/abs/2010.13166

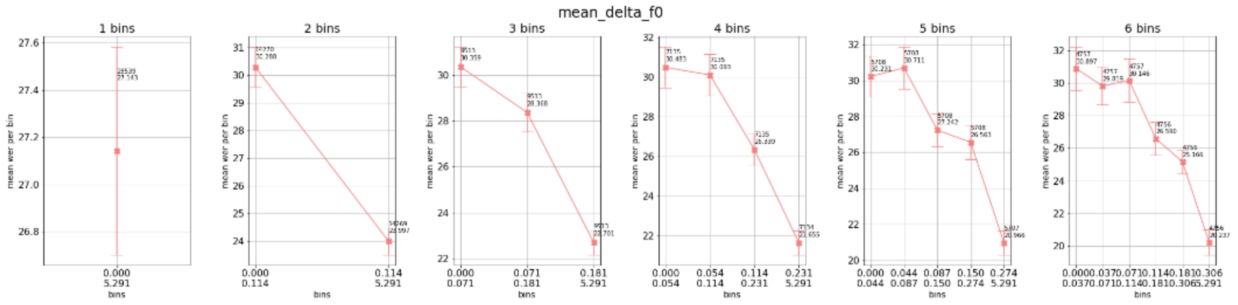
Wittig, F., & Uller, C. M. (2003). Implicit Feedback for User-Adaptive Systems by Analyzing the Users' Speech. *Proceedings of the Work-Shop on Adaptivit ̈at Und Benutzermodellierung in InteraktivenSoftwaresystemen (ABIS)*. http://www.praat.org

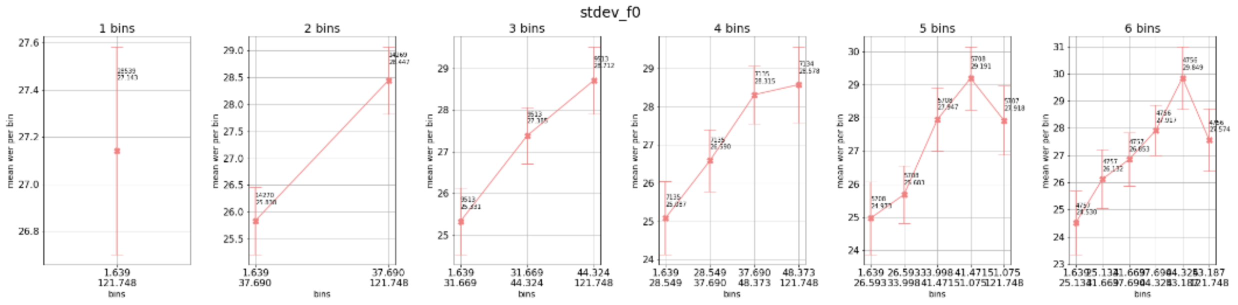
Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., Platen, P. von, Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. le, Gugger, S., ... Rush, A. M. (2020). *Transformers: State-of-the-Art Natural Language Processing*. 38–45. https://doi.org/10.18653/V1/2020.EMNLP-DEMOS.6

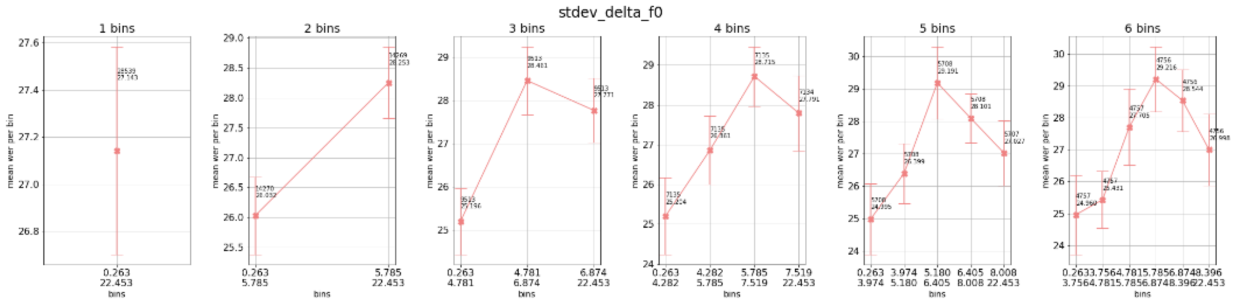
Zhang, W., Chang, X., Qian, Y., & Watanabe, S. (2020). Improving end-to-end single-channel multi-talker speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *28*, 1385–1394.

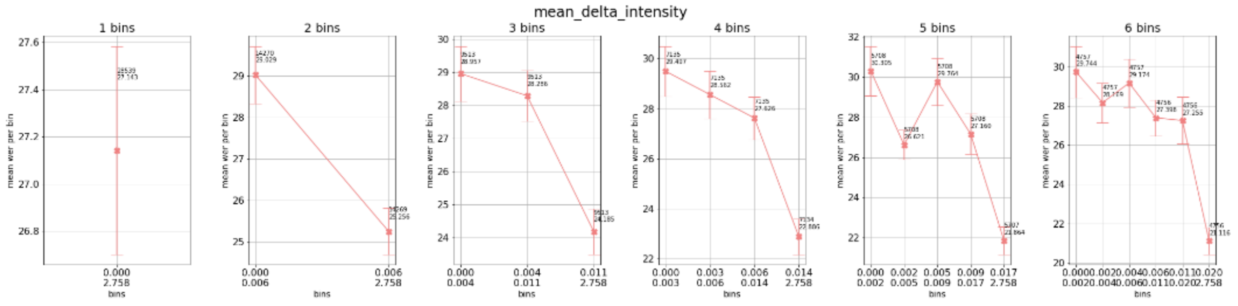
Annex: Partitions of prosodic features



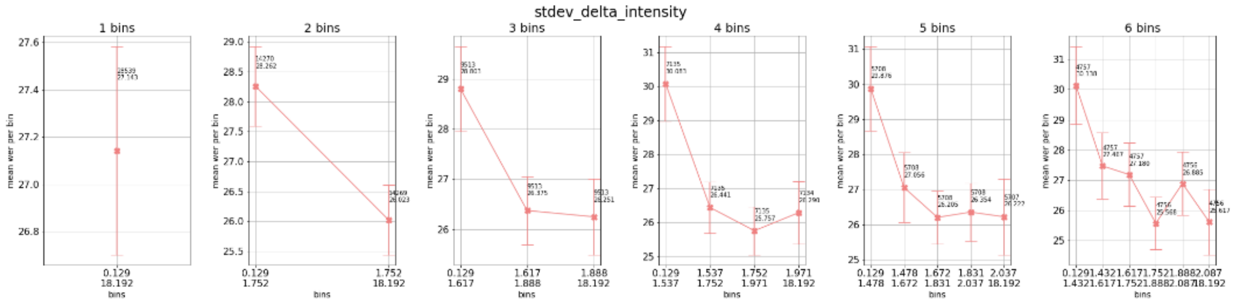




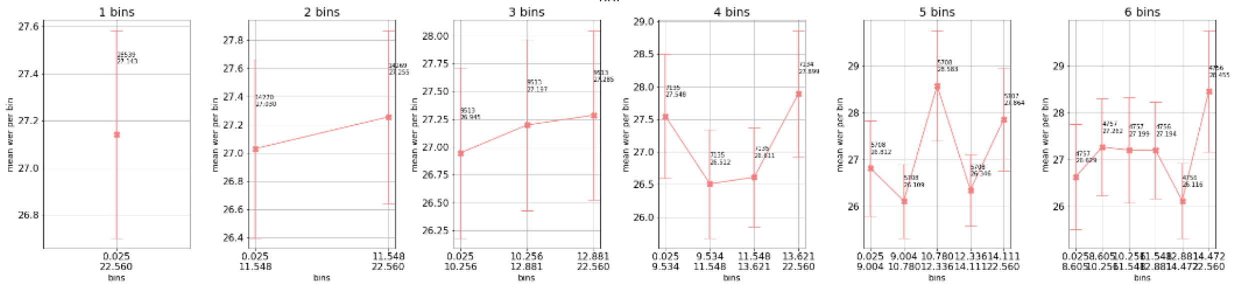




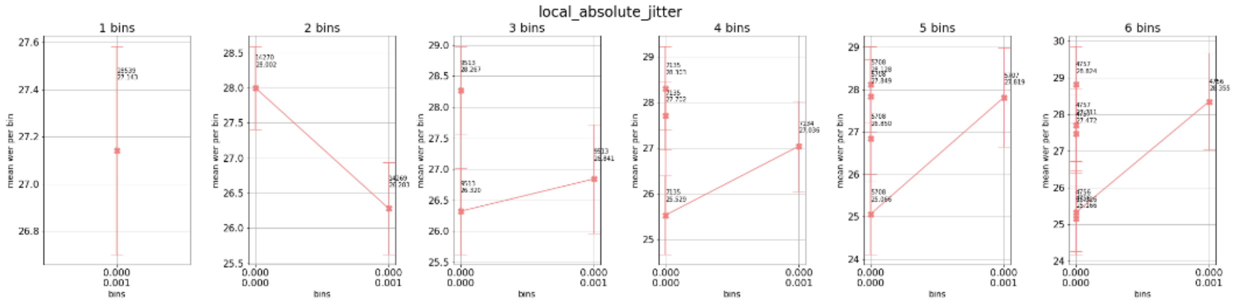
stdev_delta_intensity



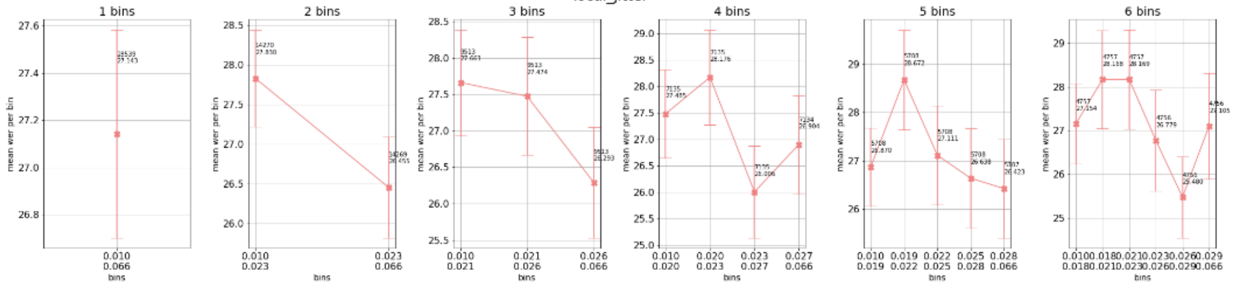
hnr



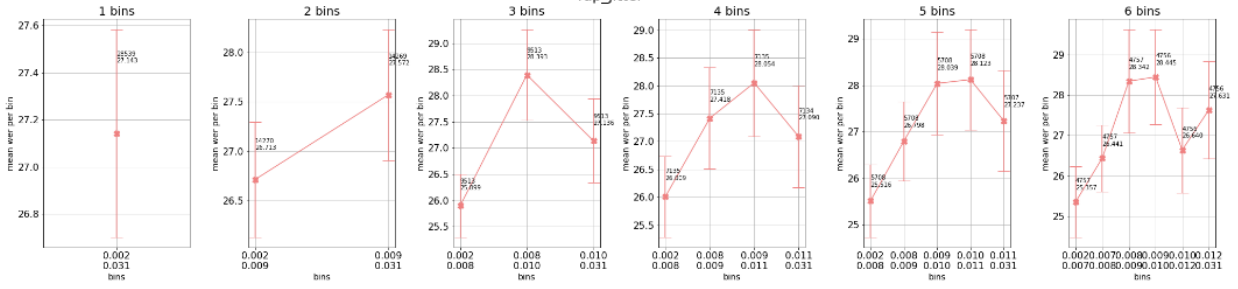
local_absolute_jitter

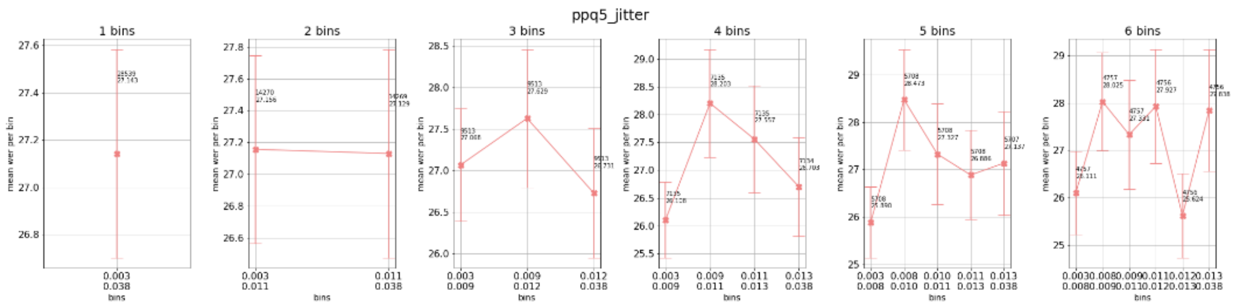


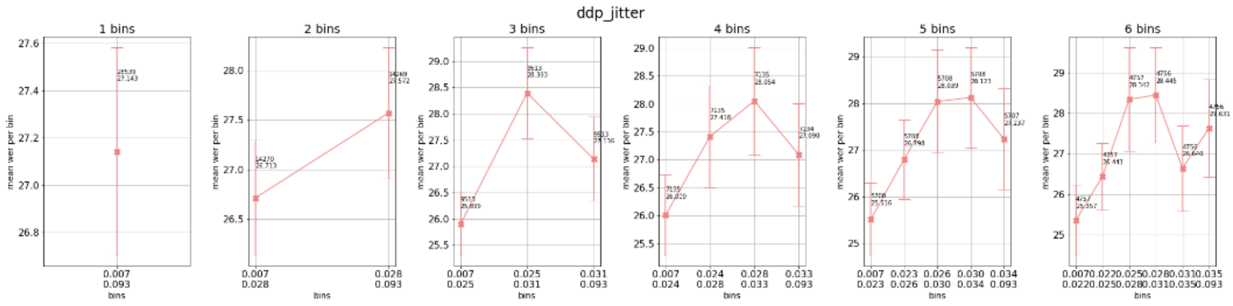
local_jitter

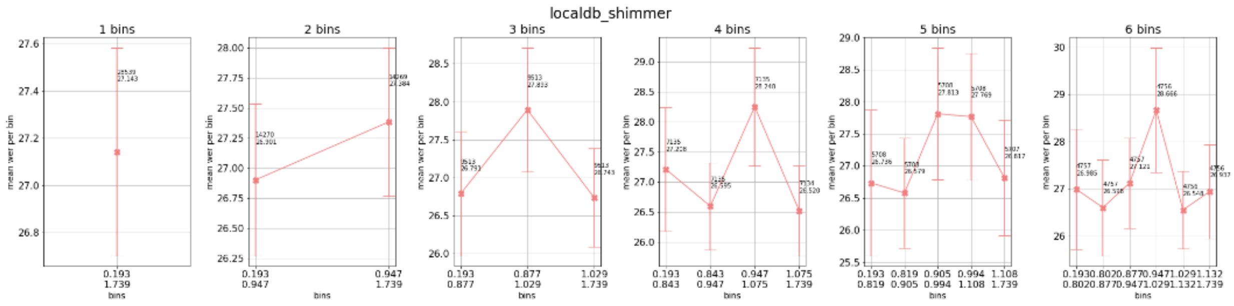


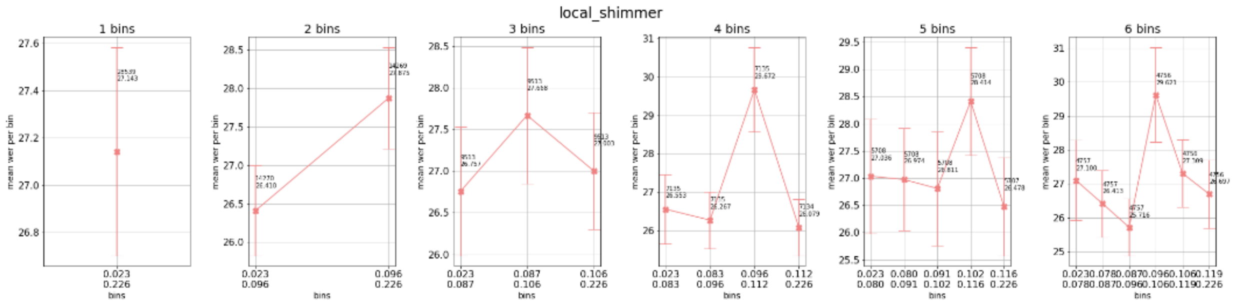
rap_jitter

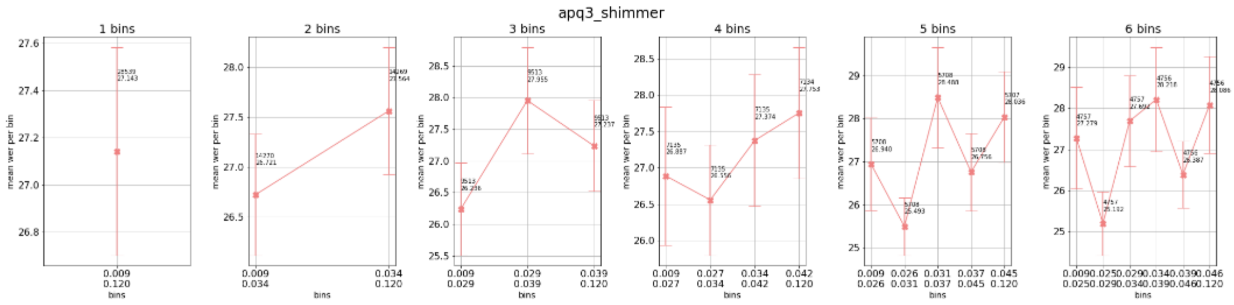


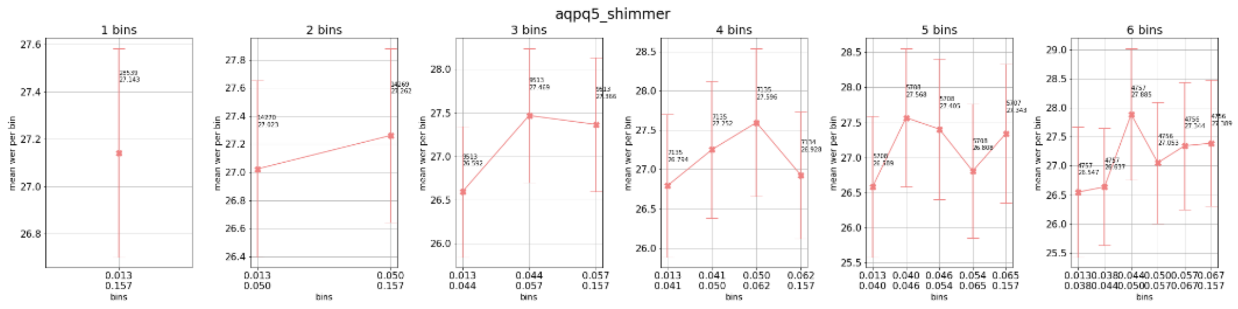

ppq5_jitter
1 bins
2 bins
3 bins
4 bins
5 bins
6 bins
mean wer per bin
bins

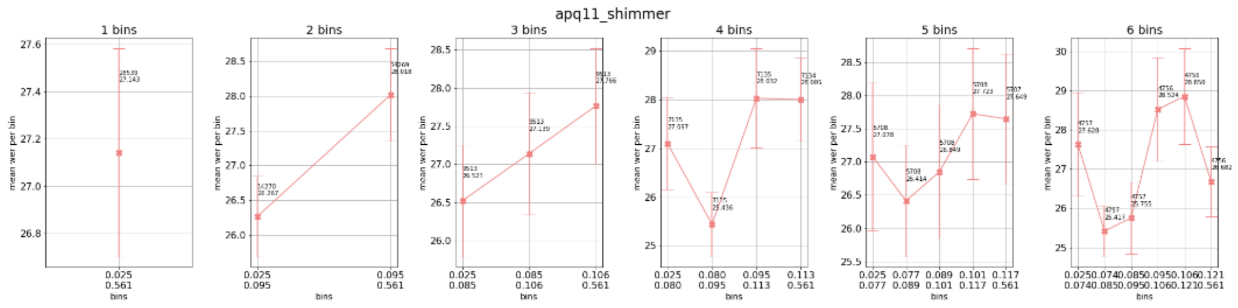

ddp_jitter
1 bins
2 bins
3 bins
4 bins
5 bins
6 bins
mean wer per bin
bins

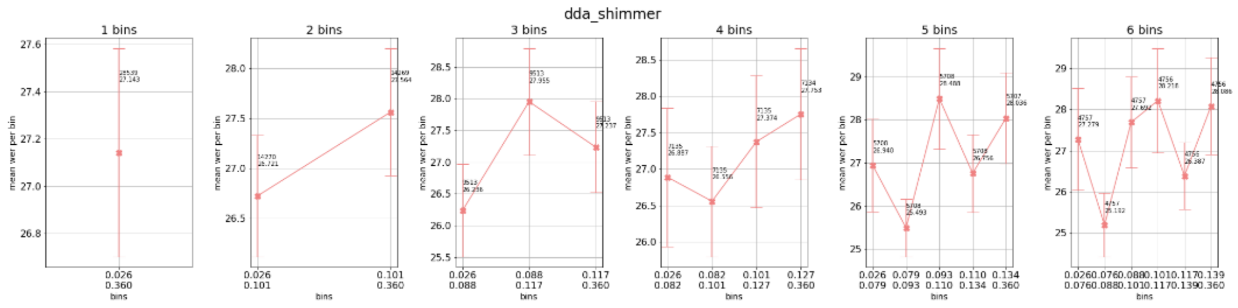

localdb_shimmer
1 bins
2 bins
3 bins
4 bins
5 bins
6 bins
mean wer per bin
bins


local_shimmer
1 bins
2 bins
3 bins
4 bins
5 bins
6 bins
mean wer per bin
bins


apq3_shimmer
1 bins
2 bins
3 bins
4 bins
5 bins
6 bins
mean wer per bin
bins


aqpq5_shimmer
1 bins
2 bins
3 bins
4 bins
5 bins
6 bins
mean wer per bin
bins


apq11_shimmer
1 bins
2 bins
3 bins
4 bins
5 bins
6 bins
mean wer per bin
bins


dda_shimmer
1 bins
2 bins
3 bins
4 bins
5 bins
6 bins
mean wer per bin
bins