Applying the conformal prediction paradigm for the uncertainty quantification of an end-to-end automatic speech recognition model (wav2vec 2.0)

Fares Ernez FARESERNEZ@GMAIL.COM
Alexandre Arnold ALEXANDRE.ARNOLD@AIRBUS.COM
Audrey Galametz AUDREY.GALAMETZ@AIRBUS.COM
Catherine Kobus CATHERINE.KOBUS@AIRBUS.COM
Nawal Ould-Amer NAWAL.OULD-AMER@AIRBUS.COM

Editor: Harris Papadopoulos, Khuong An Nguyen, Henrik Boström and Lars Carlsson

Abstract

Uncertainty quantification is critical when using Automatic Speech Recognition (ASR) in High Risk Systems where safety is highly important. While developing ASR models adapted to such context, a range of techniques are being explored to measure the uncertainty of their predictions. In this paper, we present two algorithms: the first one applies the Conformal Risk Control paradigm to predict a set of sentences that controls the Word Error Rate (WER) to an adjustable level of guarantee. The second algorithm uses Inductive Conformal Prediction (ICP) to predict uncertain words in an automatic transcription. We analyze the performance of the three algorithms using an open-source ASR model based on Wav2vec 2.0. The CP algorithms were trained on the "clean test" part of the LibriSpeech corpus that contains approximately 2,600 sentences. The results show that the three algorithms provide valid and efficient prediction sets. We guarantee that the WER is below 2% with a confidence level of 80% and an average set size of 29 sentences and we detect 90% of the badly transcripted words.

Keywords: uncertainty quantification, conformal prediction, natural language processing, automatic speech recognition, high risk systems, robustness

1. Introduction

Air Traffic Control (ATC) is a service provided by ground-based air traffic controllers who direct aircrafts on the ground and through given sections of controlled airspace. The primary purpose of ATC worldwide is to prevent collisions, organize and expedite the flow of air traffic, and provide information and other support for pilots. ATC may issue instructions that pilots are required to obey, or advisories that pilots may, at their discretion, disregard. The pilot in command is the final authority for the safe operation of the aircraft and may, in an emergency, deviate from ATC instructions to the extent required to maintain safe operation of their aircraft. Air traffic controllers monitor the location of an aircraft in their assigned airspace by radar and communicate with the pilots by radio using distinctive call signs.

Having an on-board ASR application to keep track of the received messages would reduce the pilot's cognitive workload. It would however certainly require the addition of robustness assessment and/or uncertainty quantification tools to ensure its reliability. The recent advances in Automatic Speech Recognition (ASR) and Natural Language Processing (NLP) technologies have certainly opened the way to potential applications in the field of aeronautics and Air Traffic Control (ATC). (Pellegrini et al., 2018)(Delpech et al., 2018)

The present work illustrates an application of the conformal prediction paradigm as an uncertainty quantification technique to an open-source end-to-end Speech-to-text model based on wav2vec 2.0 and trained on a public dataset (LibriSpeech).

The paper is organised as follows. We will start by introducing Conformal Prediction as well as wav2vec 2.0, the Automatic Speech Recognition model used in this study. We will then present two applications of conformal prediction: the first aims to predict a set of sentences with a certain guarantee on the error and the second one aims to predict the uncertain words in an automatic transcription.

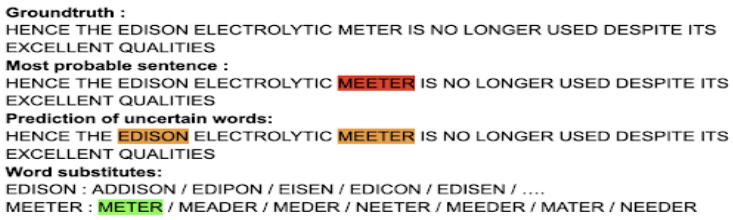


Figure 1: Example

2. Related Work

Conformal prediction has scarcely been used in NLP tasks. (Maltoudoglou et al., 2020) applied inductive conformal prediction (ICP) on a transformers based model for the task of sentiment classification. Paisios et al. (2019) also investigated the use of ICP for the task of multi-label text classification. Dey et al. (2021) used ICP for the task of text infilling and part-of-speech prediction for natural language data. To our knowledge, conformal prediction has not been applied yet to the field of Automatic Speech Recognition.

3. Conformal Prediction

3.1. Introduction

Conformal Prediction (CP) is a set of methods designed to evaluate the uncertainty of predictions produced by any machine learning (ML) model. Learning from a calibration set, CP enables the creation of statistically rigorous intervals in regression problems or a set of classes in classification problems. The prediction sets are valid in a distribution-free sense: they provide explicit, non-asymptotic guarantees even without distributional assumptions or model assumptions.

3.2. Basic Setting and Assumptions

Consider a basic setting from (N. Balasubramanian et al.) where there is no ML model involved.

Given a set of examples, the goal is to predict a new example. We will assume that the examples are elements of a measurable space $\mathbf{Z}$ with $\mid \mathbf{Z} \mid > 1$. The examples of the set will be denoted $z_1, ..., z_l$ and the one to be predicted z_{l+1}.

We will make two assumptions about how the examples $z_1, ..., z_{l+1}$ are generated:

- **Randomness** : It assumes that the $l+1$ examples are generated independently from the same probability distribution Q on Z
- **Exchangeability** : It assumes that the sequence $(z_1, ..., z_{l+1})$ is generated from a probability distribution P on Z^{l+1} that is exchangeable i.e for any permutation π of the set $\{1, ..., l+1\}$,

the predicted sequence $(z_{\pi(1)}, ..., z_{\pi(l+1)})$ is generated from the same probability distribution P on Z^{l+1}

A CP algorithm constructs a set predictor which is a function Γ that maps any sequence $(z_1, ..., z_l)$ to a set $\Gamma(z_1, ..., z_l) \subseteq Z$. Such set is called a prediction set. The statement implicit in a prediction set is that it contains z_{l+1} and it is regarded as erroneous if it fails to contain z_{l+1}.

The two main indicators of a set predictor are its validity and its efficiency. A set predictor is exactly valid at a significance level α if $P(z_{l+1} \notin \Gamma(z_1, ..., z_l)) = \alpha$. Efficiency measures the informativeness of a prediction set. An example of a prediction set we would better avoid is the whole example set Z, it is absolutely reliable but not informative.

A trade-off needs to be found between reliability and informativeness depending on the significance level α.

The last object to define is the non-conformity measure $A : Z \to \mathbb{R}$. It assigns to every element of the example space a non-conformity score.

The conformal predictor Γ defined by A as a non-conformity measure and α as a significance level is defined by: $\Gamma^\alpha(z_1, ..., z_l) := \{z | p^z > \alpha\}$ where for each $z \in Z$, the corresponding p-value p^z is defined by $p^z = \frac{|\{i \in [1,l+1], \alpha_i^z \geq \alpha^z\}|}{l+1}$ and $\forall i \in [1, l+1], \alpha_i = A(z_i)$.

Under the exchangeability and randomness assumptions, the probability of error will not exceed α because an error is made if and only if α_{l+1} is among the $\lfloor \alpha(l+1) \rfloor$ largest elements in the sequence $(\alpha_1, ..., \alpha_{l+1})$. Because of the exchangeability assumption, all permutations of $(\alpha_1, ..., \alpha_{l+1})$ are equiprobable and a random permutation moves one of the $\lfloor \alpha(l+1) \rfloor$ largest elements to the $(l+1)th$ posititon with probability α which is therefore the probability of error. This proposition was proved in (Vovk et al.).

3.3. Inductive Conformal Prediction for classification

We are now given an example space $Z = X \times Y$ where X the object space and $Y = \{y_1, ..., y_k\}$ the label space. Unlike the full conformal prediction framework that uses the whole dataset to predict a new data point, Inductive or Split Conformal Prediction (ICP) learns a set predictor on a fixed calibration dataset. It learns on less data points and is therefore more computationally efficient.

Algorithm 1: Inductive Conformal Prediction for a Classification Problem

Require: Dataset $\{(X_i, Y_i)\}_{i=1}^n$, significance level α

1: Split the dataset into 3 substets I_{train}, I_{calib} and I_{test}
2: Train a ML model $\hat{f}$ on I_{train}
3: Define the non-conformity measure $A(X, Y) \in \mathbb{R}$
4: Compute $\hat{\lambda}$ as the $\frac{\lfloor (n_{calib}+1)(1-\alpha) \rfloor}{n_{calib}}$ quantile of the calibration scores $\{s = A(X, Y), \forall (X, Y) \in I_{calib}\}, n_{calib} = |I_{calib}|$
5: Use this quantile to form the prediction set for a new examples. $\Gamma^\alpha(X_{test}) := \{Y \in \{Y_1, ..., Y_k\} | A(X_{test}, Y_k \leq \hat{\lambda}\}$
6: Evaluate the conformal set predictor validity and efficiency on I_{test}

The validity of the prediction sets is guaranteed for any non-conformity measure and distribution of the data by the Conformal coverage guarantee theorem.

Theorem 1 (Conformal coverage guarantee; Vovk, Gammerman, and Saunders) *Suppose $(X_i, Y_i)_{i=1,...,n}$ and (X_{test}, Y_{test}) are independent and identically distributed. $\hat{q}$ is defined as in step 4 above and $\Gamma^\alpha(X_{test})$ as in step 5 above. Then the following holds: $P(Y_{test} \in \Gamma^\alpha(X_{test}) \geq 1 - \alpha$*

Proof *See Appendix D of [9]* ■

3.4. Conformal Risk Control

The Conformal Risk Control is an extension of the basic Conformal Prediction framework to provide guarantees of the form $E[l(\Gamma(X_{test}), Y_{test})] \leq \alpha$ for any bounded loss function l that shrinks as the prediction set grows.

Theorem 2 (Conformal Risk Control) *Consider a set predictor $\Gamma_\lambda(.)$ that depends on a parameter λ that encodes its level of conservativeness and a loss function $l(\Gamma(X), Y) \in (-\infty, B]$ bounded by $B < \infty$ and non-increasing as a function of λ. Let $\hat{R}_n(\lambda) = \frac{\sum_{i=1}^{n_{calib}} l(\Gamma_\lambda(X_i), Y_i)}{n_{calib}}$ and $\hat{\lambda} = \inf\{\lambda | \frac{n_{calib}}{n_{calib}+1}\hat{R}_n(\lambda) + \frac{B}{n_{calib}} \leq \alpha\}$ with α the desired significance level.*

The set predictor $\Gamma_{\hat{\lambda}}$ guarantees that $\alpha - \frac{2B}{n_{calib}+1} \leq E[l(\Gamma(X_{test}), Y_{test})] \leq \alpha$.

Proof *See theorem 1 and 2 in (N. Angelopoulos et al.).* ■

Note that the coverage guarantee seen in the past paragraph is a special case of Conformal Risk Control with $l(\Gamma^\alpha(X), Y) = \mathbb{1}(Y \notin \Gamma(X))$

3.5. Effect of the calibration set (Training Conditional Validity)

In the past paragraphs, we tackled the validity of conformal predictors in the sense of unconditional validity while it is possible to explore their conditional validity, among other things (label conditional, object conditional, ...) and on the calibration dataset (Training-conditional validity).

The property of training-conditional validity (TCV) has been formalized in (Vovk) using a PAC-type 2-parameters definition which means that for an example space $Z = \{X_i, Y_i\}_{i=1..n}$ in which the samples are iid, a set predictor is $(\alpha, \delta) - valid$ if for any probability distribution P on Z, $P^{n_{calib}}(E(l(\Gamma^\alpha(X), Y) \leq \alpha) \geq 1 - \delta$.

Proposition 2b in (Vovk) states that such predictor is $(\alpha, \delta) - valid$ if and only if the nonconformity measure is continuous and $\delta \geq bin_{n,\alpha}(\lfloor \alpha(n+1) - 1 \rfloor)$.

Proposition 2a in (Vovk) states that a set predictor $\Gamma^{\alpha - \sqrt{\frac{-ln(\delta)}{2n}}}$ will be $(\alpha, \delta) - valid$. The term $\sqrt{\frac{-ln(\delta)}{2n}}$ can be seen as a correction to the significance level that makes the set predictor less conservative to take into account the effect of the calibration dataset. It is recommended to set $\delta = 0.1$ in the literature. (N. Angelopoulos and Bates)

3.6. Label-conditional and feature-conditional Inductive Conformal Prediction

The motivation behind Conditional Conformal Predictors comes from the fact that ICPs do not always achieve the required probability α of error $Y_{l+1} \notin \Gamma^\alpha(X_{l+1})$ conditional on $(X_{l+1}, Y_{l+1}) \in E$ for important sets $E \subseteq \mathbf{Z}$.

An inductive m-taxonomy is a measurable function $K : Z^m \times \mathbf{Z} \to \mathbf{K}$ where $\mathbf{K}$ is a measurable space. Usually the category $K((z_1, ..., z_m), z)$ of an example z is a kind of classification of z, which may depend on the proper calibration set $(z_1, ..., z_m)$.

The conditional inductive conformal predictor for a classification task Γ defined by A as a nonconformity measure and α as a significance level is defined by: $\Gamma^\alpha(z_1, ..., z_l, x) := \{y \mid p^y > \alpha\}$ where for each $y \in Y$, the corresponding p-value p^y is defined by $p^y = \frac{|\{i \in [1,l], \kappa_i = \kappa^y \wedge \alpha_i \geq \alpha^z\}| + 1}{|i \in [1,l], \kappa_i = \kappa^y| + 1}$ with $\forall i \in [1, l], \alpha_i = A(z_i), \kappa_i = K(., z_i)$ and $\kappa^y = K(., (x, y))$.

A label-conditional ICP is a conditional ICP with the m-taxonomy $K(., (x, y)) = y$ and a feature-conditional ICP is a conditional ICP with the m-taxonomy $K(., (x, y)) = x_i$ with $i \in [1, m]$ if the object space X can be divided into m subsets.

Theorem 3 *If random examples $Z_1, ..., Z_l, Z_{l+1} = (X_{l+1}, Y_{l+1})$ are exchangeable, the probability of error $Y_{l+1} \notin \Gamma^\alpha(Z_1, ..., Z_l, X_{l+1})$ given the category $K((Z_1, ..., Z_l), Z_{l+1})$ of Z_{l+1} does not exceed α for any α and any conditional inductive conformal predictor Γ corresponding to K.(Vovk)*

3.7. Evaluating Conformal Prediction

We start with plotting the histograms of the prediction set sizes. We benefit from these histograms in two ways. First, a big average set size would suggest that the conformal process is not particularly precise, suggesting that there may be an issue with the score or underlying model. Second, the distribution of the prediction set sizes reveals whether the prediction sets correctly adapt to the difficulty of examples. A broader spread is typically preferred because it indicates that the method is successfully differentiating between easy and difficult inputs.

The next step is to verify if the theoretical coverage guarantee has been achieved and to compute the empirical converage conditionally on certain features or labels that could be of interest to explore possible miscoverages on certain groups of the example space and correct it by implementing the Conditional ICP.

Running the algorithm once will not be enough to check the validity of our conformal predictor. We will be running the procedure over R trials, resampling the calibration and test sets at each trial.

Let $C_j = \frac{1}{n_{val}} \sum_{i=1}^{n_{val}} \mathbb{1}\{Y_{i,j}^{(val)} \in \Gamma_j^\alpha(X_{i,j}^{(val)})\}$, for $j = 1, ..., R$

We will plot a histogram of the $(C_j)_{j=1..R}$ and verify that it is centered at roughly $1-\alpha$.(N. Angelopoulos and Bates)

4. Automatic Speech Recognition (ASR) and wav2vec 2.0

Wav2vec 2.0 (Baevski et al., 2020) is a framework for semi-supervised learning of representations from both labeled and unlabeled audio data. The training was done in two phases: a pre-training phase using self-supervised learning to achieve the best speech representation possible, and a fine-tuning phase that uses labeled speech data to learn to predict sequences of words. The model pre-training procedure makes it more frugal in terms of need of annotated data for the speech recognition downstream task. By pre-training on 53k hours of unlabeled data and only using ten minutes of labeled speech data, the model achieves a 4.8% WER on the LibriSpeech clean test set.

4.1. Wav2vec 2.0 pretraining mechanism

The architecture of the model consists of a multi-layer convolutional feature encoder $f : \mathbf{X} \rightarrow \mathbf{Z}$ which takes as input raw audio $\mathbf{X}$ and outputs latent speech representations $\mathbf{z} = z_1, \ldots, z_T$ for $\mathbf{T}$ time steps.

The model makes use of a Quantization Module to automatically learn discrete speech units to benefit from the fact that voiced speech could be separated into phones. The Quantization Module discretizes the output of the feature encoder z to a finite set of speech representations via Product Quantization. The idea of Product Quantization is to decompose the space into a Cartesian product of low dimensional subspaces and to quantize each subspace separately. The space of $\mathbf{Z}$ has been decomposed into G codebooks or groups, each one consisting of V code words or entries. The feature vector z is then mapped to $l \in \mathbb{R}^{G \times V}$ logits. One entry is chosen for each group using a hard Gumbel softmax.

$$v^* = \underset{v}{\text{argmax}} \frac{exp((l_{g,v} + n_v)/\tau)}{\sum_{k=1}^{V} exp((l_{g,k} + n_k)/\tau)}$$

With $n_v = -log(log(u))$, u a random sample from $U(0, 1)$ and τ the temperature. n_v and τ make the difference compared to the usual softmax function. This adds randomization to encourage the model to use different entries during training.

The entries are concatenated to form a vector of quantized speech representations $\mathbf{q}$ of the same size of $\mathbf{c}$. We naturally get a vector of quantized speech representations for each time step and end up with $\mathbf{q} = q_1, \ldots, q_T$.

Similarly to masked language modeling in BERT(Devlin et al., 2019), certain time steps in the output of the feature encoder are masked to the Transformer module during pre-training and the objective function that is minimized is a sum of a contrastive loss L_m and a diversity loss L_d.

$$L = L_m + \alpha L_d$$

For each masked time step t, the contrastive loss' objective is to train the model to predict a vector representation c_t similar to the true quantized latent speech representation q_t. A set of K+1 candidates Q_t which contains q_t and K distractors is used:

$$L_m = -log \frac{exp(sim(c_t, q_t))/\tau}{\sum_{\tilde{q} \in Q_t} exp(sim(c_t, \tilde{q})/\tau)}$$

The diversity loss is a regularization technique, its objective is to encourage the model to take advantage of all code words.

$$L_d = \frac{1}{GV} \sum_{g=1}^{G} \sum_{v=1}^{V} \frac{exp((l_{g,v} + n_v)/\tau)}{\sum_{k=1}^{V} exp((l_{g,k} + n_k)/\tau)}$$

4.2. Model fine-tuning

The fine-tuning phase requires labeled data i.e., pairs of audios and their corresponding ground-truth text. A recurrent neural network is trained to take as input the vector of context representations $\mathbf{c} = c_1, \ldots, c_T$ and output a matrix containing a score for each token from a list of 30 tokens, for each time-step. A token can be a character, a combination of characters, a word boundary token or a blank token, not to be confused with a white space.

The model is trained by minimizing a Connectionist Temporal Classification (CTC) loss (Graves et al.). It is usually used to train models for sequence to sequence problems where we don't have the alignment between the input and the output. It is the case for speech recognition; we do not know the alignment of the characters in the corresponding audio. For example, if "hello" was transcripted without a CTC loss, we could get "hhhheeeellllllllooooo". Removing all duplicates would result in "helo". The CTC loss solves the issue by learning to predict the blank token at the right place.

4.3. Beam Search decoding

A fine-tuned model based on wav2vec 2.0 uses the beam search method to decode a CTC-output matrix. It comes down to iteratively create text candidates (beams) and score them.

To compute the logit score of a beam, we compute the sum of the log-probabilities of all the paths that lead to the beam. For example, over 3 time-steps, the score of "aa" is given by:

$$P_{t3}(aa) = P_{t1}(a).P_{t2}(a).P_{t3}(-) + P_{t1}(a).P_{t2}(-).P_{t3}(a) + P_{t1}(-).P_{t2}(a).P_{t3}(a)$$

with (-) the blank token and $P_{t2}(a)$ corresponds to the score of the token "a" at the 2nd timestep in the CTC-output matrix.

An optional language model can be used at the end of the beam search to compute a language model score for every decoded sentence. The lm-score can be either added to or averaged to the logit score to give the combined score.

A simplified pseudo-code of a modified version of wav2vec 2.0's beam search that will output not only one but a set of sentences and their corresponding scores is detailed in figure 2

Algorithm 2: Modified version of the Beam Search (wav2vec 2.0)

1: $beams \leftarrow$ [] Empty list
2: $scores \leftarrow \{\}$ Empty dictionary that assigns a score for each beam
3: **for** i in range(T) **do**
4: $\quad bestBeams \leftarrow bestBeams(beams, scores, BW, beamPruneLogp)$
5: $\quad beams \leftarrow$ [] Empty list
6: $\quad$ **for** b in bestBeams **do**
7: $\quad\quad$ **for** c in tokens **do**
8: $\quad\quad\quad$ **if** $mat(index(c), t) \geq tokenMinLogp$ **then**
9: $\quad\quad\quad\quad path \leftarrow concat(b, c)$
10: $\quad\quad\quad\quad scores[(path, t)] \leftarrow calcScore(mat, path, t)$
11: $\quad\quad\quad\quad beams \leftarrow beams \cup path$
12: $\quad\quad\quad$ **end if**
13: $\quad\quad$ **end for**
14: $\quad$ **end for**
15: $\quad beams, scores \leftarrow mergeBeams(beams, scores)$
16: **end for**
17: **if** useLanguageModel = TRUE **then**
18: $\quad beams, scores \leftarrow addLanguageModelScore(beams, scores)$
19: **end if**
20: **return** beams, scores

The list of beams and a score dictionary are initialized with an empty list (step 2 and 3). The algorithm iterates over the time-steps (step 4).

At each time-step, only the best scoring beams from the previous time-step are kept and are sorted in descending order based on their scores (step 5). Beam width (BW) specifies the maximum number of beams to keep and beamPruneLogp specifies the maximum difference between the score of the best beam and the last one to keep.

Further, each beam is extended by the tokens whose logit score is superior to tokenMinLogp (step 9 and 10) and a logit score is calculated for each path (step 11). After that, the paths that lead to the same beam are merged, the score is recalculated and the beams are sorted (step 16).

An optional language model score (also a log-probability) is computed for each beam and is added to the logit score of the beam and the beams are sorted (step 18).

Note that the number of predicted sentences per audio depends on the audio and the parameters (BW, tokenMinLogp, beamPruneLogp) but can only be lower than BW.

4.4. Word Error Rate

The Word Error Rate (WER) is a common metric used to assess the performance of an ASR model. It is computed as : $WER = \frac{S+D+I}{N} = \frac{S+D+I}{S+D+C}$ where S is the number of substitutions, D is the

number of deletions, I is the number of insertions, C is the number of correct words and N is the number of words in the reference ($N = S + D + C$).

A WER of 5-10% is considered to be good quality. Cloud-based Speech-To-Text services like Amazon, Google and Microsoft for example claim to achieve a WER of approximately 5%. We note that multiple benchmarkings have refuted this claim and showed that their average WER may be closer to 10% than 5%. (Jarmulak; Xu et al.)

We will use the Word Error Rate metric to measure the precision of our model as it is more informative than the exact transcriptions ratio. Nonetheless, we acknowledge that WER has strong flaws e.g., it does not consider the fact that some words are more important for the general meaning of the sentence than others.

5. Data

We used the model 'wav2vec2-base-100h-with-lm' from Huggingface (hug) which was trained on 53k hours of unlabeled data and finetuned using LibriSpeech's (lib) 'train clean' subset (around 100h of labeled speech). It uses a 4-gram language model (a 4-state Markov chain) to compute the language model score, trained with KenLM (Heafield). We used the 'test clean' subset which contains 2620 audios of different durations (4h20min in total) to implement the algorithms used in the next chapters.

First we give some insights about this dataset. Fig. 2 shows the distribution of the number of time-steps or tokens in the the CTC-output matrices of the 2620 audios. It can be seen as the distribution of the durations of the audios given the fact that a time step is approximately $20ms$.

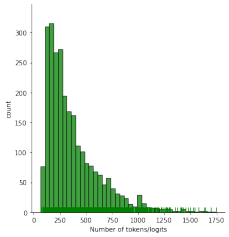


Figure 2: Distribution of the number of tokens/logits per audio

In order to evaluate the effect of the beam search's parameters on the computational time and the number of sentences that would be predicted, we picked three audios that have approximately the same number of time-steps as the (0.5,0.75,0.95)-quantiles and predicted their outputs using different sets of parameters (BW, tokenMinLogp, beamPruneLogp).

	number of time-steps	number of sentences predicted	time (s)
$\approx 0.5 - quantile$	97	114	0.54
$\approx 0.75 - quantile$	154	99	0.84
$\approx 0.95 - quantile$	286	76	2.1

Table 1: BW = 200, tokenMinLogp = -10, beamPruneLogp = -100 (set one)

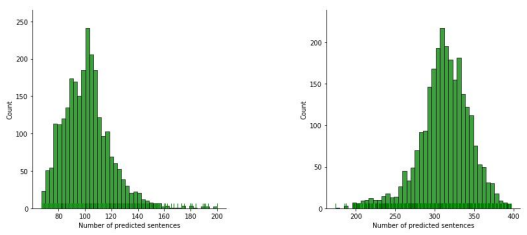
	number of time-steps	number of sentences predicted	time (s)
$\approx 0.5 - quantile$	97	287	5.91
$\approx 0.75 - quantile$	154	323	10
$\approx 0.95 - quantile$	286	311	19.2

Table 2: BW = 400, tokenMinLogp = -15, beamPruneLogp = -150 (set two)

We chose to use the parameters of the sets one and two in order to compare them as it was a good trade-off between computational time and number of sentences predicted. The goal was to make the same experiments with the two sets to see which one has the best 'value for money'. We will name the two datasets we obtain "Economy" dataset (set one) and "Quality" dataset (set two).

	"Economy" dataset	"Quality" dataset
mean $n_{sentences}$	102	312
std $n_{sentences}$	18	33
max $n_{sentences}$	200	397
min $n_{sentences}$	67	174
exact first transcription	60.53%	61.57%
exact prediction in set	82.21%	84%
mean minWer	1.57%	1.36%
std minWer	5.19%	4.6%
mean bestWerIndex	2.6	4.8
std bestWerIndex	8.4	16.7
max bestWerIndex	109	174

Table 3: Statistics of Datasets "Economy" and "Quality"



(a) "Economy" dataset (b) "Quality" dataset

Figure 3: Histograms of number of predicted sentences per audio

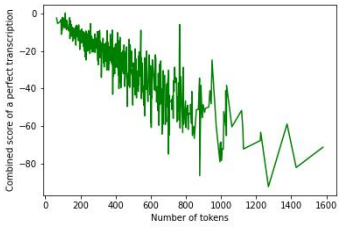


Figure 4: Combined score of transcriptions identical to the ground truth versus number of tokens for the 'Economy' dataset

Fig. 4 that shows the transcription score versus the token number. We see that the longer the sentences, the larger the scores (in absolute value). This comes as no surprise as the scores are accumulated along the word hypotheses. These scores therefore need to be normalized if we want to use them as non-conformity scores.

6. Applying the Conformal Risk Control algorithm to predict a set of sentences

The goal is to predict, for each audio, a set of sentences that guarantees, with a certain significance level, that at least one of the sentences has a WER lower than a chosen wer_{target} . As it is not a classification nor a regression task, we resort to the Conformal Risk Control framework that is explained in 3.4. Since the output score depends on the length of the sentence, a softmax is applied on the top-k sentences and the resulting values are used to construct the set predictor.

6.1. Algorithm

Given a wer_{target}, an audio X, its ground truth transcription Y, the set of sentences predicted by the model $sent(X) = sent_1, ..., sent_{n_X}$ ranked from the most probable to the least probable, the corresponding raw output scores $score_1, ..., score_{n_X}$ with $n_X \leq BW$, k the maximum number of sentences in the conformal prediction sets, the loss function is the following:

$$l(\Gamma(X), Y) = \mathbb{1}(\forall i \in [1, n_\lambda], WER(sent_i, Y) \geq wer_{target})$$

with $n_\lambda = inf\{j \in [1, min(k, n_X)], \sum_{l=1}^{j} \frac{exp(score_l)}{\sum_{i=1}^{min(k,n_X)} exp(score_i)} \geq \lambda\}$

It is clear that the loss is bounded by $B = 1$ and that it is non-increasing as a function of λ. The higher λ is, the more sentences will be included in the conformal prediction set $\Gamma_\lambda(X)$ and so the higher the probability will be to include a sentence that has a lower WER than wer_{target}.

The conditions to build an $(\alpha, \delta) - valid$ set predictor under the Conformal Risk Control framework (3.4) are then satisfied.

We note that the wer_{target} should be higher than the $meanMinWer$ and that the confidence level $1 - \alpha$ should be lower than the proportion of audios whose top-k sentences contain at least a sentence with a lower WER than the wer_{target}.

Algorithm 3: Conformal Risk Control to predict a set of sentences prediction

Require: Dataset $\{(X_i, Y_i)\}_{i=1}^{n}$, significance level α, δ, wav2vec 2.0 parameters (BW, tokenMinLogp, beamPruneLogp), maximum number of sentences k, wer_{target}, the binary search's precision ϵ

1: Predict a set of sentences for each audio using the modified version of wav2vec 2.0 with the parameters (BW, tokenMinLogp, beamPruneLogp) to obtain a set of sentences $sentences_i = sent_{1_i}, ..., sent_{n_{X_i}}$ and their corresponding scores $scores_i = score_{1_i}, ..., score_{n_{X_i}}$ for each audio in the dataset.

2: $\forall i \in [1, n], n_{X_i} \leftarrow min(k, n_{X_i})$ We keep k sentences if $n_{X_i} > k$ and we keep all sentences if $n_{X_i} < k$

3: Apply a softmax function on the top-k scores. $\forall i \in [1, n], softmaxScores_i = (\frac{exp(score_{1_i})}{\sum_{j=1}^{n_{X_i}} exp(score_{j_i})}, ..., \frac{exp(score_{n_{X_i i}})}{\sum_{j=1}^{n_{X_i}} exp(score_{j_i})})$

4: Compute the wer array for each audio. $\forall i \in [1, n], wer_i = (wer(sent_{1_i}, Y_i), ..., wer(sent_{n_{X_i}}, Y_{n_{X_i}}))$. The dataset is now of the form $\{X_i, Y_i, sentences_i, softmaxScores_i, wer_i\}_{i=1}^{n}$

5: Verify: $wer_{target} \geq MeanMinWer$ with $MeanMinWer = \frac{\sum_{i=1}^{n} min(wer_i)}{n}$ and $\alpha \geq \alpha_{min}$ with $\alpha_{min} = \sum_{i=1}^{n} \mathbb{1}(min(wer_i) \geq wer_{target}) \times \frac{1}{n}$ If not, choose a higher wer_{target} or a smaller α

6: Split the dataset into 2 substets I_{calib} and I_{test}

7: Verify: $\delta \geq bin_{n_{calib},\alpha}(\lfloor \alpha(n_{calib} + 1) - 1 \rfloor)$ If not, augment the calibration dataset's size or α or δ

8: $\lambda_{array} \leftarrow Array(0, 1, step = \epsilon)$

9: Proceed to a binary search to find $\hat{\lambda} = \inf\{\lambda \in \lambda_{array} | \frac{n_{calib}}{n_{calib}+1} \hat{R}_n(\lambda) + \frac{B}{n_{calib}} \leq \alpha - \sqrt{\frac{-ln(\delta)}{2n}}\}$ with $\hat{R}_n(\lambda) = \frac{\sum_{i=1}^{n_{calib}} l(\Gamma_\lambda(x_i), y_i)}{n_{calib}}$, l as defined above and $B = 1$

10: Evaluate the $(\alpha, \delta) - valid$ conformal predictor on the test dataset by computing the empirical coverage $\hat{\alpha} = \sum_{(X,Y) \in I_{test}} l(\Gamma_{\hat{\lambda}}(X), Y) \times \frac{1}{n_{test}}$ and the mean size of the conformal predictions sets

6.2. Implementation

For the experiments, we adopt $\delta = 0.1$ and a precision $e = 0.0001$. We make k vary $\in \{50, 100, 300\}$, for the top-k sentences that we will keep to predict the CP sets. Each k can yield a different $MeanMinWer$. We set the target WER to $wer_{target} = 2\%$. α is put to 0.2. It would not be possible to hope for a smaller significance level due to the Training-Conditional Validity hypothesis. The following tables summarise our experiments.

	$MeanMinWer$	wer_{target}	α_{min}	α
$k = 50$	1.6%	2%	0.172	0.2
$k = 150$	1.6%	2%	0.168	0.2
$k = 300$	1.6%	2%	0.168	0.2

Table 4: Experiments with "Economy"

	$MeanMinWer$	wer_{target}	α_{min}	α
$k = 50$	1.6%	2%	0.166	0.2
$k = 150$	1.4%	2%	0.153	0.2
$k = 300$	1.4%	2%	0.152	0.2

Table 5: Experiments with "Quality"

6.3. Results

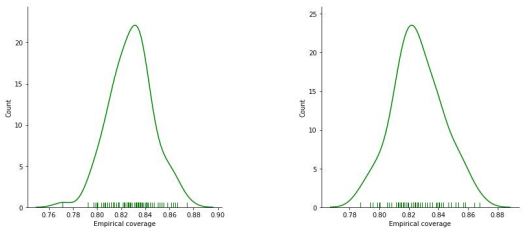
The empirical confidence level in the tables below represent the mean of the distribution of the empirical confidence levels $1 - \hat{\alpha}$ of $R = 100$ trials. (See 3.7).

	wer_{target}	theoretical confidence level	empirical confidence level	average set size
$k = 50$	2%	80%	82.38%	36
$k = 150$	2%	80%	82.82%	44
$k = 300$	2%	80%	82.5%	40

Table 6: Results with "Economy"

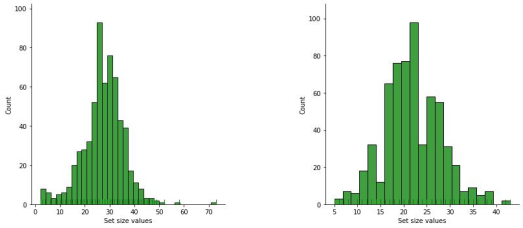
	wer_{target}	theoretical confidence level	empirical confidence level	average set size
$k = 50$	2%	80%	82.65%	31
$k = 150$	2%	80%	82.4%	30
k = 300	**2%**	**80%**	**82.45%**	**29**

Table 7: Results with "Quality"



(a) "Economy" dataset (b) "Quality" dataset

Figure 5: Distribution of the empirical coverage for k = 150



(a) "Economy" dataset (b) "Quality" dataset

Figure 6: Distribution of the prediction set sizes for k = 150

- The empirical coverage lower bound is respected for all cases. The fact that it is higher than the upper bound is due to the implementation of the training-conditional validity.
- It seems that using "Quality" dataset with k = 50 yields the most adaptive prediction sets with the same validity. This confirms the intuitive idea that more time and storage consuming implementations of the beam search give better and more precise sentences.
- Different variants of the non-conformity measure were tried (applying a softmax on the raw output scores, applying a softmax after dividing the score by the number of time steps of the audio..) but the best results were observed when applying a softmax after dividing the output scores by 5.

- We could investigate feature-conditional ICP over the number of tokens (> 0.5-quantile or not) because shorter audios tend to yield better transcriptions. See 6.5

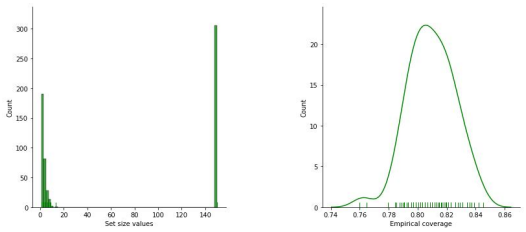
6.4. Examples

Figure 7: Example of a CP set of sentences

Outputting a set of 30 sentences can appear unpractical but this model can be coupled with other models to improve their efficiency.

6.5. Example of feature-conditional ICP

We arbitrarily use "Quality" dataset with k = 150, $\text{wer}_{target} = 2\%$ and $\alpha = 0.2$. The setup is similar to the previous section but using feature-conditional ICP on the number of tokens/logits of the audio. We divide the dataset in two subsets; $subset_0$ contains audios whose number of tokens is inferior to the 0.5-quantile and $subset_1$ contains audios whose number of tokens is superior to the 0.5-quantile. We implement the Risk Control Framework independently. Results are shown in Fig. 8.



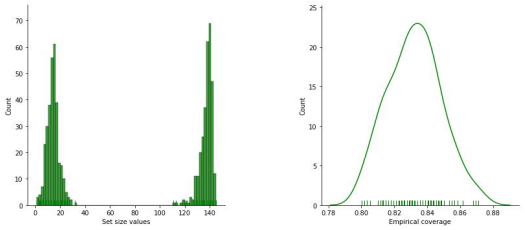
(a) Distribution of the prediction set sizes

(b) Distribution of the empirical coverage

Figure 8: Results of the feature-conditional ICP when not adapting α

A simultaneous gain (probably for $subset_0$) and loss ($subset_1$) of adaptivity can be observed. This may be due to the fact that wer_{target} and α are not appropriate for the two subsets.

We decide to adapt wer_{target} and α to the two subsets. We take $wer_{target_0} = wer_{target_1} = 2\%$, $\alpha_0 = 0.15$ and $\alpha_1 = 0.25$ and obtain the results presented in Fig. 9.



(a) Distribution of the prediction set sizes

(b) Distribution of the empirical coverage

Figure 9: Results of the feature-conditional ICP when adapting α

We show that it is possible to improve adaptivity with the assumption that $P(ntokens_{newexample} > 0.5 - quantile) = P(ntokens_{newexample} < 0.5 - quantile)$.

7. Applying the Conformal Prediction algorithm to predict uncertain words in a sentence

Motivated by the fact that a long list of sentences are difficult to digest for humans, we want to further explore guarantees at a world level on the top-1 sentence using CP. The idea is to predict the wrongly transcribed words in the most probable sentence using a set of sentences that can be defined by the top-k sentences or by the conformal predictions sets introduced in 6. It could be done by training a classifier that predicts whether each word of the most probable sentence is correct or not using features extracted from the set of sentences.

To train such a classifier, we will need to construct a dataset $\{word, feature_1, ..., feature_x, label\}_{i=1}^{n}$

7.1. Dataset construction

In this section we explain how the dataset for "uncertain words" is constructed. A word from the most probable sentence will be labeled 1 (correct) if it is present at the same position than in the ground truth transcription (with a tolerance of 1 word to account for possibly different sentence lengths) The score represents the presence rate of the corresponding word in the sentences of the set. This score heuristic was inspired from the example 7 where the wrongly transcribed word is predicted differently in nearly each sentence of the set.

Consider the following input:

- most probable sentence : HE IS AN ENGINEER
- set of sentences :
  - sentence 1 : HE IS AND ENGINEER
  - sentence 2 : HE ISN'T AN ENGINEER
  - sentence 3 : HE IS A PILOT
  - sentence 4 : HE IS AND ARCHITECT
- groundtruth sentence : HE IS AN ENGINEERING TEACHER

We would obtain the following dataset:

	firstWord	lastWord	nChar	nWords	scoreWBefore	scoreWAfter	score	label
HE	1	0	2	4	Nan	0.75	1	1
IS	0	0	2	4	1	0.25	0.75	1
AN	0	0	2	4	0.75	0.5	0.25	1
ENGINEER	0	1	8	4	0.25	Nan	0.5	0

Table 8: Example of dataset construction

In the example above, the word "HE" is present in 4 out of 4 sentences. Thus, it has a score of 1 and since it is the first word of the sentence, the score of the word before (scoreWBefore) is not attributed. The word "IS" is present in 3 out of 4 sentences and has a score of 0.75. nChar matches the word's number of characters and nWords matches the number of words in the sentence.

7.2. Algorithm

Based on the dataset presented in last section, we implement the Inductive Conformal Prediction algorithm introduced in §2.3 . We split the dataset and train an XGBoost classifier $\hat{f}(x : [score, numCharacters, numWordsInSentence, firstWord, ...] \rightarrow y : label$. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework.(xgb)

We set the non-conformity measure to be one minus the softmax output of the true class. It is the ordinary non-conformity measure for a classification task. The non-conformity score will be high when the model is not confident in the output of the true class and vice versa.

A slight correction of the significance level will be made in the algorithm 1 in §2.3 to take into account the effect of the calibration dataset.

The set predictor will predict whether a word is correct $\{1\}$, wrong $\{0\}$ or uncertain if it outputs $\{1, 0\}$ at a confidence level of $1 - \alpha$ with a probability $1 - \delta$.

7.3. Implementation

As in 6, we will make experiments on both datasets "Economy" and "Quality".

	"Economy" dataset	"Quality" dataset
$n_{elements}$	52780	52773
number of wrong words (label = 0)	2915	2893
number of correct words (label = 1)	49865	49880
ratio n_0/n_1	0.058	0.058
mean score of correct words	0.91	0.91
std score of correct words	0.16	0.13
mean score of wrong words	0.64	0.65
std score of correct words	0.29	0.29

Table 9: Statistics of Datasets "Economy" and "Quality" for uncertain words prediction

The resulting datasets are very unbalanced in the number of wrong/correct words.

Splitting the dataset and Training the XGBoost Classifier We propose the following split of the "Quality" dataset with k=150:

	$n_{sentences}$	n_{words}	n_0	n_1
Entire dataset	2620	52773	2893	49880
XGBoost training	1400	28725	1603	27122
XGBoost testing	400	8663	317	8346
Calibration	500	9343	591	8752
Test	320	6041	381	5659

Table 10: Split of "Quality" dataset with k = 150

We train the XGBoost classifier by assigning the ratio n_0/n_1 to the "scale_pos_weight" parameter. It makes sure that the model gives more weight to the misrepresented labels during training.

Other parameters: n_estimators = 300, learning_rate = 0.3

The results of the classifier on the test dataset are detailed in the following tables

Accuracy	True Positive Rate (TPR)	True Negative Rate (TNR)	Balanced Accuracy
0.88	0.89	0.44	0.67

Table 11: Classifier's performance

In the table above 11, $BalancedAccuracy = (TPR + TNR)/2$

For the experiments, we use $\delta = 0.1$ whenever training-conditional validity (TCV) is used, varying $k \in \{50, 100, 300\}$ and making $R = 100$ trials to compute the empirical coverage. We will also be varying $\alpha \in \{0.01, 0.05, 0.1\}$.

7.4. Results

We obtain the following results:

	mean set size	$coverage_{marginal}$	$coverage_0$	$coverage_1$
$\alpha = 0.1$ with TCV	**1.1**	**0.9123**	**0.54**	**0.94**
$\alpha = 0.05$ with TCV	1.33	0.9615	0.72	0.98
$\alpha = 0.01$ with TCV	NaN	NaN	NaN	NaN
$\alpha = 0.1$ without TCV	1.06	0.9	0.51	0.93
$\alpha = 0.05$ without TCV	1.27	0.9516	0.67	0.97
$\alpha = 0.01$ without TCV	1.68	0.9898	0.88	1

Table 12: Results with "Economy" dataset (k = 150)

	mean set size	$coverage_{marginal}$	$coverage_0$	$coverage_1$
$\alpha = 0.1$ with TCV	1.08	0.9108	0.51	0.94
$\alpha = 0.05$ with TCV	1.32	0.9601	0.65	0.98
$\alpha = 0.01$ with TCV	NaN	NaN	NaN	NaN
$\alpha = 0.1$ without TCV	1.06	0.9001	0.49	0.93
$\alpha = 0.05$ without TCV	1.25	0.949	0.6	0.97
$\alpha = 0.01$ without TCV	1.63	0.9895	0.87	1

Table 13: Results with "Quality" dataset (k = 150)

- The average set size indicates the average number of words that would be flagged as uncertain by the CP-algorithm. For a mean set size of 1.28, 28% of the words would be predicted as uncertain.
- We obtain slightly more adaptive prediction sets (smaller mean set size) when using "Quality" dataset but it is maybe not worth a 10x computation cost in practice.
- As it was expected, smaller significance levels yield higher prediction set sizes.
- The lower bound of the coverage is respected for the experiments with TCV.
- The lower and higher bounds for the marginal coverage are not always respected for the experiments without TCV although they are very close ($O(\frac{1}{n_{calib}})$) which seems alright since the theoretical lower and upper bounds stand when testing on an infinite dataset.
- It is not possible to implement the Training-Conditional Validity with $\alpha = 0.01$ with a calibration dataset of 9343 elements.
- The coverage for correct words (label 1) is higher than the one for incorrect words. It was expected given the misrepresentation of incorrect words (20x less). This leads to trying label-conditional ICP (See section 7.5).

7.5. Example of label-conditional ICP

We present here an example of label-conditional ICP. We used "Quality" dataset with k = 150 and did the same experiments as in 4.2.3. We obtain the following results:

	mean set size	$coverage_{marginal}$	$coverage_0$	$coverage_1$
$\alpha = 0.1$ with TCV	1.68	0.91	0.94	0.91
$\alpha = 0.1$ without TCV	1.55	0.9	0.9	0.9
$\alpha = 0.05$ with TCV	1.9	0.96	0.99	0.96
$\alpha = 0.05$ without TCV	1.82	0.97	0.97	0.97
$\alpha = 0.01$ with TCV	NaN	NaN	NaN	NaN
$\alpha = 0.01$ without TCV	1.9	0.99	0.99	0.99

Table 14: Results of label-conditional ICP with "Quality" (k = 150)

We show that it is possible to have an equally-valid conformal predictor but with the risk of obtaining large predictions sets and consequently losing efficiency. This is due to the fact that the dataset lacks 0-labeled elements (wrongly transcribed words). This shows the importance of applying label-conditional ICP when it is critical to have the same guarantee on every label.

7.6. Examples

The examples below were generated using "Economy" dataset (k = 150) and with TCV, we used two different values for α to highlight the effect of the significance level on the prediction of uncertain words. The wrongly transcribed words are highlighted in red in "Most probable sentence" and the uncertain words are highlighted in orange in "Prediction of uncertain words".

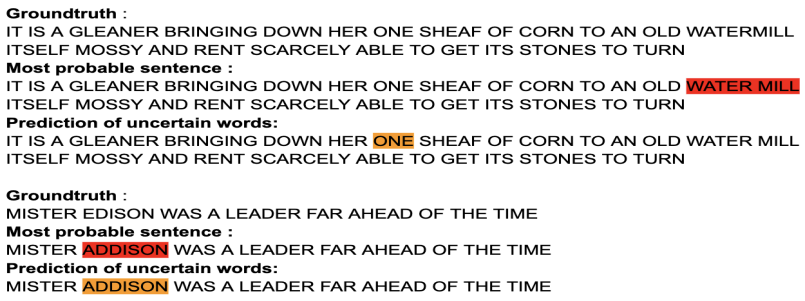


Figure 10: $\alpha = 0.1$ with TCV

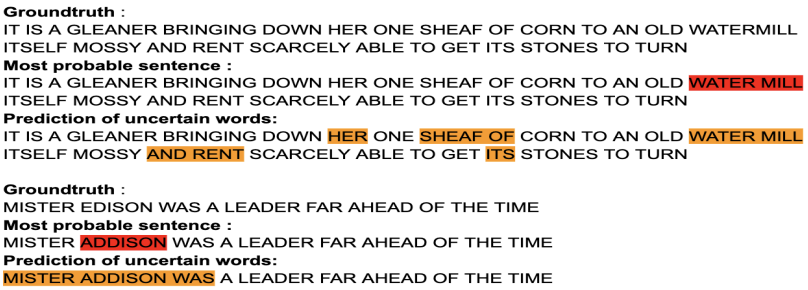


Figure 11: $\alpha = 0.05$ with TCV

8. Conclusion

This works presents a promising application of Conformal Prediction in the uncertainty quantification of an Automatic Speech Recognition model and the results are quite encouraging. We guaranteed a 2% WER with a confidence a level of 80% by predicting a CP set of 30 sentences in average. We guaranteed a coverage of 90% for the prediction of uncertain words while being uncertain on 10% of the words.

This work also shows that a less time and storage-consuming way of extracting sentences ("Economy" dataset) happens to be more efficient than the "greedy" way ("Quality" dataset) for uncertain words prediction. However, for the prediction of a set of sentences, "Quality" dataset gives the most efficient conformal predictor (10 fewer sentences in average in the conformal prediction sets).

Given that only four hours of audio data were used to train, calibrate and test the conformal predictors, we can hope to achieve better results if we used more data. Indeed, we expect that more data would enable us to implement label-conditional and feature-conditional conformal prediction or to ask for a very small significance level.

The next step will naturally consist on testing these algorithms on the Air Traffic Control Data.

For further development, we intend to investigate other metrics than the Word Error Rate, other scoring functions and other ways of computing p-values that could yield more informative prediction sets. We could use Deep Learning methods to improve the predictors.

The computational and storage cost of such models could also be investigated from the angle of Embedded AI.

References

huggingface. https://huggingface.co/patrickvonplaten/wav2vec2-base-100h-with-lm.

librispeech. https://www.openslr.org/12/.

xgboost. https://xgboost.readthedocs.io/en/stable/.

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *CoRR*, abs/2006.11477, 2020. URL https://arxiv.org/abs/2006.11477.

Estelle Delpech, Marion Laignelet, Christophe Pimm, Céline Raynal, Michal Trzos, Alexandre Arnold, and Dominique Pronto. A Real-life, French-accented Corpus of Air Traffic Control Communications. In *Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan, May 2018. URL https://hal.science/hal-01725882.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Neil Dey, Jing Ding, Jack Ferrell, Carolina Kapper, Maxwell Lovig, Emiliano Planchon, and Jonathan P Williams. Conformal prediction for text infilling and part-of-speech prediction, 2021.

Alex Graves, Santiago Fernandez, Faustino Gomez, and Jurgen Schmidhuber. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. URL https://www.cs.toronto.edu/~graves/icml_2006.pdf.

Kenneth Heafield. KenLM Language Model Toolkit. URL https://kheafield.com/code/kenlm/.

Jacek Jarmulak. https://www.voicegain.ai/post/speech-to-text-accuracy-benchmark-june-2022. URL https://www.voicegain.ai/post/speech-to-text-accuracy-benchmark-june-2022.

Lysimachos Maltoudoglou, Andreas Paisios, and Harris Papadopoulos. Bert-based conformal predictor for sentiment analysis. In Alexander Gammerman, Vladimir Vovk, Zhiyuan Luo, Evgueni Smirnov, and Giovanni Cherubin, editors, *Proceedings of the Ninth Symposium on Conformal and Probabilistic Prediction and Applications*, volume 128 of *Proceedings of Machine Learning Research*, pages 269–284. PMLR, 09–11 Sep 2020. URL https://proceedings.mlr.press/v128/maltoudoglou20a.html.

Anastasios N. Angelopoulos and Stephan Bates. A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification. URL https://people.eecs.berkeley.edu/~angelopoulos/publications/downloads/gentle_intro_conformal_dfuq.pdf.

Anastasios N. Angelopoulos, Stephan Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal Risk Control. URL https://arxiv.org/pdf/2208.02814.pdf.

Vineeth N. Balasubramanian, Shen-Shyang Ho, and Vladimir Vovk. *Conformal Prediction for Reliable Machine Learning - Theory, Adaptations and Applications.*

Andreas Paisios, Ladislav Lenc, Jiří Martínek, Pavel Král, and Harris Papadopoulos. A deep neural network conformal predictor for multi-label text classification. In Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, and Evgueni Smirnov, editors, *Proceedings of the Eighth Symposium on Conformal and Probabilistic Prediction and Applications*, volume 105 of *Proceedings of Machine Learning Research*, pages 228–245. PMLR, 09–11 Sep 2019. URL https://proceedings.mlr.press/v105/paisios19a.html.

Thomas Pellegrini, Jérôme Farinas, Estelle Delpech, and François Lancelot. The airbus air traffic control speech recognition 2018 challenge: towards ATC automatic transcription and call sign detection. *CoRR*, abs/1810.12614, 2018. URL http://arxiv.org/abs/1810.12614.

Vladimir Vovk. Conditional validity of inductive conformal predictors. URL http://proceedings.mlr.press/v25/vovk12/vovk12.pdf.

Voldya Vovk, Alex Gammerman, and Craig Saunders. Machine-Learning Applications of Algorithmic Randomness. URL https://eprints.soton.ac.uk/258960/1/Random_ICML99.pdf.

Binbin Xu, Tao Chongyang, Youssef Raqui, and Sylvie Ranwez. A Benchmarking on Cloud based Speech-To-Text Services for French Speech and Background Noise Effect. URL https://hal.science/hal-03874256/document.