Computational Linguistics in the Netherlands Journal 12 (2022) 19-32 Submitted 12/2022; Published 12/2022

Design Principles of an Automatic Speech Recognition Functionality in a User-centric Signed and Spoken Language Translation System

Aditya Parikh* ADITYA.PARIKH@RU.NL
Louis ten Bosch* LOUIS.TENBOSCH@RU.NL
Henk van den Heuvel* HENK.VANDENHEUVEL@RU.NL
Cristian Tejedor-García* CRISTIAN.TEJEDORGARCIA@RU.NL

Radboud University, Nijmegen, The Netherlands

Abstract

The European project SignON aims at designing a user-oriented and community-driven platform for communication among deaf, hard of hearing, and hearing individuals in both sign language and spoken languages (i.e. English, Dutch, Spanish, and Irish). Inclusion, easy access to translation services and the use of state-of-the-art Artificial Intelligence (AI) are the key aspects of the platform design. Users can communicate to the system with text via typing, speech via a microphone, and sign language through video, while the system can respond using, for instance, (translated) output, subtitles, translated audio via speech synthesis, and via a 3D avatar. In this framework, the design of a flexible, user-friendly component for Automatic Speech Recognition (ASR) is a challenge, due to the constraints imposed by the platform, in terms of usability and the flexible use of system-external services. This paper addresses the current state-of-the-art ASR component in SignON and the conceptual choices underlying the design, operation, and integration of the ASR component in the SignON application.

1. Introduction

Access to information is generally considered everyone's right. In this globalized world, inclusive access is often characterized by exchange of multilingual content and by cross-lingual communication. In order to facilitate this communication, it is important to cross the language barrier in order to support accessibility by technological applications and scientific improvement. The SignON project is an example. It aims to research and develop a spoken and sign language (SL) recognition and translation based mobile application in an open communication framework. The main goal of the SignON project is to create a link for communication among deaf, hearing, and hard-of-hearing people.

In this paper we present an overview of the status of automatic speech recognition (ASR) functionality in the SignON project, describing the milestones and various approaches to address the challenges and peculiarities of ASR. We will also discuss the methodology to create and deploy the ASR model and how it functions in the SignON project. We investigate the performance of ASR in terms of word error rate on a number of audio corpora. In the Discussion section, we present future steps in ASR within the SignON project.

2. About the ASR in the SignON project

The primary objective of the SignON project is to develop a machine translation system for hearing and hard-of-hearing people by creating Sign and Spoken recognition models of various languages in all possible combinations as shown in the figure 1. One of the key aspects of the project is the scalability of the developed system towards multiple languages. Currently, the project is based on

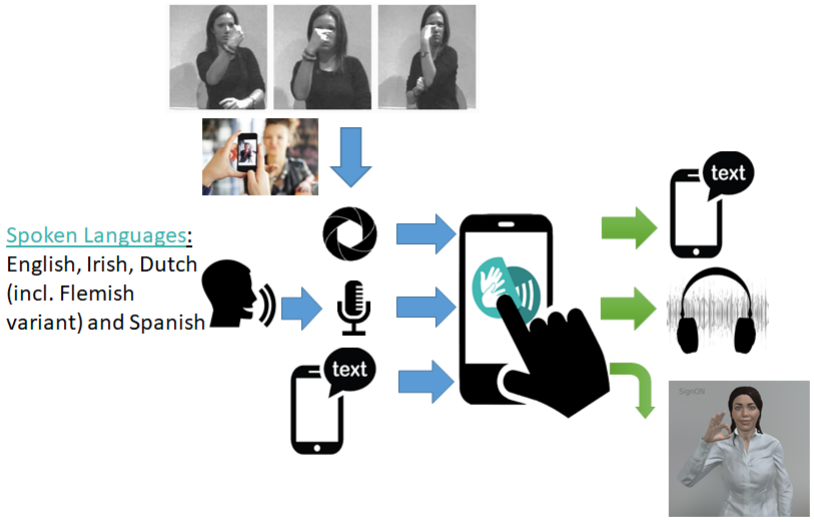

Figure 1: Overview of the modular set-up of the SignON application. Each module is related to a scientific part in the entire multidisciplinary SignON project. The input for the application is text, audio (speech), or video (sign language). The output is a translated version of the input (e.g., other languages, other modalities, or both). In the middle of the picture, we have the end user's tablet/smartphone, which may connect to external dedicated auxiliary translation services via internet.

the use of four languages: English, Spanish, Irish, Dutch, and Flemish (henceforth Belgian Dutch). While for most of these languages sufficient audio and text material is available, Irish is an exception. The lack of audio and text material for Irish inspired a different training approach for this language based on transfer learning.

The ASR component in SignON is tuned to the use cases and to the particular type of speaker. This design includes the use of atypical speech from deaf speakers and/or speakers with cochlear implants. The ASR application also addresses the challenges of latency: for a smooth operation of the SignON application and improved user satisfaction, the system latency should be as low as possible and the output modalities should as much as possible be in sync with each other and with the input. Finally, the ASR component in the SignON project adheres to the European General Data Protection Regulation (GDPR) in terms of data storage, logging and restrictions on audio data access.

2.1 Approaches to creating the ASR

A conventional general hybrid automatic speech recognition (ASR) system is based on an integration of an acoustic model (AM), a dictionary and a language model (LM), in combination with a word search module. These modules can be trained and tuned independently (see below). In the SignON system, the acoustic models are based on a deep neural network (DNN) for estimating the posterior probability of each phoneme state given a specific spectral frame of the input speech signal. The phoneme states are defined by the Hidden Markov Model (HMM) models used for the description of each phone. The language models are created from text corpora by means of the SRILM toolkit

(Stolcke 2002). The ASR models for English, Spanish, Dutch and its Flemish variant are created with Kaldi ASR (Povey et al. 2011), while for Irish we used Wav2Vec2 (Baevski et al. 2020) for developing an end-to-end ASR system via finetuning a language-independent model XLS-R (Babu et al. 2021) from HuggingFace[1].

2.2 ASR Webservice Description

In the SignON application, the functionality of the ASR component is implemented via a webservice named audioserver (abbreviated AS) (Tejedor-García 2022). AS is a state-of-the-art backend web-service for transcribing (decoding) audio files with ASR technology in real-time via standard https requests. AS is multilingual, and all languages integrated within SignON are supported by AS.

AS facilitates easy deployment and development, by combining Docker, UNIX/Linux, Node.JS Express, MongoDB, and the Kaldi Automatic Speech Recognition system. As seen from the AS webservice, the SignON application is a client. As a client, it can send/receive information to/from the AS through a JSON REST API. This guarantees a flexible and scalable ASR infrastructure, including the possibility of selecting the language, acoustic models, and a search beam parameter on-the-fly for decoding. The beam search parameter determines the balance between recognition accuracy and decoding latency of the Kaldi ASR system. The web server keeps track of all user's interactions with the system using a non-relational database. Regarding data security, the interaction protocol is based on https for encrypted and secure client-server data transmission and uses JSON Web Token (JWT) for login authentication and secure requests. Audio files sent to the server are checked before being uploaded (the checking is with respect to file media type and size). Besides, it is possible to put a limit on the number of requests per specific user. For additional security, the audio files can be (automatically) removed at server side after obtaining the transcription (the user can select this option on-the-fly). In addition, the REST API developed offers a secure login procedure and JWT security to the https requests and establishes "rules" of communication between the client apps and the server in JSON format (e.g., maximum file size, audio file format, etc.). Appendix 1 in section 7.1 is showing a reference script and the output about how the AS makes the connections at the client side on the SignON platform.

The latency of the ASR web service is a critical feature that is important for the usability of the entire system and for user satisfaction. Table 1 presents the latency results obtained so far with a client (beam parameter = 15) and a standard wi-fi connection. The real-time factor (RTF) decreases from 0.180 to 0.147 when decoding audio files from 5 seconds to 102 seconds in length, respectively. This solution is characterized by a very low latency at the webservice side (less than 15% RTF). The overall response time on the level of the SignON application will depend on internet connection speed (between client and server), the latencies due to syncing SignON output, and the ASR-internal latencies (note that the beam parameter also plays a vital role in the latency time).

2.3 Issues addressed by AS

The challenges addressed by the ASR component in the Audioserver (AS) in the SignON application are summarized below. These points are relevant for the usability of the speech modality in the entire SignON application.

- Privacy issues (GDPR). For ASR, privacy relates to the security of the audio data, in particular the removal of the audio data from temporal storage at the server side after having completed the transcription. This storage option is up to the user. The audioserver has a setting in which very long wave files are deleted anyway after decoding, to save disk space.

1. https://huggingface.co/facebook/wav2vec2-xls-r-300m

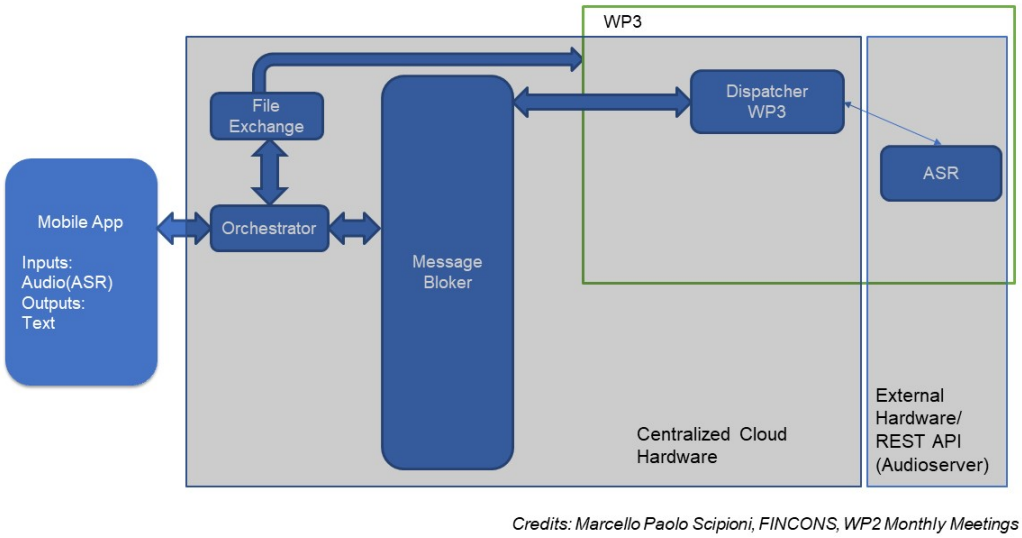

Figure 2: Functional overview of the connection between the Mobile App, the central SignOn orchestrator and the external ASR webservice.

Table 1: Latencies (measured in terms of system time) averaged over several tens of audio files. (Client: Postman Windows app connected to eduroam wifi network - Ping 11ms. Download speed: 60 Mbps. Upload speed 32.6 Mbps. Server: CLST/Lightning server + Docker + Dutch ASR AM/LM models.) The latency includes the uploaded audio file and the ASR real-time decoding with beam=15. RTF denotes real-time factor.

Audio file duration (seconds)	Latency (seconds)	RTF
5	0.8	0.160
7	1.3	0.186
10	1.8	0.180
20	3.5	0.175
40	6.0	0.150
75	11	0.147
102	15	0.147

- Flexibility of the ASR infrastructure. As observed above, the language models and acoustic models and beam parameters can be chosen on-the-fly prior to and in between decoding tasks. That is, this language/model change is also possible in between subsequent wav file uploads.

- Scalable and multilingual ASR. There is no limitation on the total number of ASR decoding languages since it is possible to add as many language/acoustic models as desired.

- Security issues. The client-server https connection goes via JWT. Passwords are encrypted using bcrypt. The total number of unsuccessful login attempts is strictly limited. Audio files

are checked (type and size) before being processed by AS. The maximum number of daily requests per user to the server can also be set.

- Latency issues. In its current set-up, there is a very low latency, to some extent dependent on the length of the audio file, the size of the ASR model, and the speed of the network connection.

3. Methods and experiments

3.1 Data

A regular training of an acoustic model requires hundreds of hours of speech utterances and reference transcriptions. For the language models, a large text corpus is required. For the SignON project ASR models, the speech and text data are mainly selected from open-source datasets. An overview is provided below.

3.1.1 Speech and Text Data

1. English ASR
   The Gigaspeech corpus (Chen et al. 2021) is used for the training of acoustic models in English ASR. Gigaspeech is an evolving, multi-domain English speech recognition corpus with 10,000 hours of high-quality labelled audio suitable for supervised training. The transcribed audio is first collected from audiobooks, podcasts, and YouTube, covering both read and spontaneous speaking styles. The Gigaspeech corpus is subdivided by the number of hours as XL (10000 hours), L (2500 hours), M(1000 hours), S (250 hours), and XS (10 hours). An 'S' small size of 250 hours of audio data is used for the training of the acoustic model. We also trained the acoustic models with 'M' (1000 hours) and 'L' (2500 hours) sizes of the Gigaspeech corpus although they did not show any significant improvement in the WER. For language models, Google's One Billion Word Benchmark (Chelba et al. 2013) text corpus is used. This corpus has almost one billion words of training data and it is mainly used for statistical language modelling. For the testing of ASR, we used Gigaspeech dev and test corpus that consist of 12.5 hours and 40 hours of audio data. Apart from that, for a fair comparison with other English ASRs, we tested the ASR on Librispeech corpus Test Clean and Others which is having 5.4 hours and 5.1 hours of audio data respectively. In the Librispeech Test corpus, lower-WER speakers were designated as 'clean' and the higher-WER speakers were designated as 'other' (Panayotov et al. 2015).

2. Spanish ASR
   For the Spanish ASR combined speech data from Commonvoice Spanish (Ardila et al. 2020), and Voxforge Spanish (Voxforge.org n.d.) is used for the acoustic modelling. The common voice dataset comes with rich metadata that includes speaker age, accent, and gender. In total 214001 utterances of Commonvoice Spanish and 3568 utterances of Voxforge Spanish were taken for training. That is 322.35 hours of speech material. For the language model, an open-source Spanish Billion Words Corpus (Cardellino 2019) is used. This Spanish text corpus consists of an unannotated corpus of the Spanish language of nearly 1.5 billion words, compiled from different corpora and resources from the web. For testing, we used Commonvoice Spanish dev and test set. Commonvoice Spanish dev set is having 26.1 hours of speech and the Commonvoice Spanish test set is having 25.9 hours of speech.

3. Flemish ASR
   The Spoken Dutch Corpus (Corpus Gesproken Nederlands, CGN) (Oostdijk 2000) is used for Flemish ASR. It is a manually orthographically annotated speech corpus of around 900 hours of contemporary Dutch speech, originating from Flemish and Dutch people. Out of 900 hours,

270 hours correspond to Southern (Flemish) Dutch. For acoustic model training, all narrow-band telephone speech and spontaneous conversational speech, which correspond respectively to components C,D, and component A of CGN were excluded. For Flemish ASR, the remaining 155 hours of audio are randomly divided into two sets: CGN-train, which contains 90% of the audio, and CGN-dev, which contains 10% of the audio. For language models, the ESAT 2008 N-best n-gram language model was used (Demuynck et al. 2009) without further adaptations. The training material for the LM was obtained from two sources: the Dutch publisher PCM (360 million words, media and newspaper texts) and the Flemish Mediargus (1,436 million words) (Kessens and van Leeuwen 2007).

4. Dutch ASR
   Similar to the Flemish ASR, the acoustic model audio data were taken from the Spoken Dutch Corpus (Corpus Gesproken Nederlands, CGN) (Oostdijk 2000). For acoustic model training of Dutch ASR, we used all the components of CGN corpus which was around 483 hours of speech. Apart from that for the language models, text corpus from CGN and Twents Nieuws Corpus (530 million words)(Ordelman et al. 2007) was used. For testing of Dutch ASR, we considered some speech from CGN corpus spontaneous speech (Dev_S) and telephonic conversations (Dev_T) which is around 0.5 hours and 1.97 hours of speech. Apart from that, we also considered Commonvoice Dutch test set in which the speech was specifically tagged as Netherlands' Dutch for testing. Those were 3512 utterances which account for 4.4 hours of speech data.

5. Irish ASR
   For Irish, an under-resourced language, public spoken audio and text datasets are hardly available. On the Commonvoice Irish (Ardila et al. 2020) 4 hours of verified audio data are available. Apart from that, the so-called 'living audio dataset' (Braude et al. 2019) now contains 1 hour of the Irish dataset. The living audio corpus is a "crowd-built" continuously growing speech dataset with transcripts. From the Commonvoice dataset, all the validated audio clips and all the living audio clips were taken into account and after a random train-test split, 90% of the total dataset were taken for training, and the rest of the 10% of real data was taken for testing. The total training data accounts for 5156 utterances which are around 5 hours of speech and for testing 579 utterances were considered that accounts for 0.5 hours of speech data.

3.2 Language and Pronunciation Models

The language model in the Kaldi hybrid system is an n-gram model trained on all text corpora listed in Section 3.1.1 using the SRILM toolkit (Stolcke 2002) for English, Dutch, Flemish, and Spanish. In general, the larger the language model (LM) that is employed, the more accurately the search lattice will be generated. Because of the limitation of memory in the SignOn platform, we performed language model pruning. The prune option in SRILM prunes N-gram probabilities if their removal causes (training set) perplexity of the model to increase by less than the threshold relative (Stolcke 2000). The words for the language models were chosen based on the frequency of words in the raw text materials. All the words below a certain threshold frequency were replaced with 'unknown word' (<unk>) tokens. For English, we used Sequence-to-Sequence G2P toolkit from cmusphinx, on the basis of the dictionary cmudict with 135k English entries. The G2P tool performs Grapheme-to-Phoneme (G2P) conversion using a transformer model from a tensor2tensor toolkit (Kaiser 2017). The transformer model deliberately avoids using recurrence and depends completely on an attention mechanism to create global dependencies between input and output (Vaswani et al. 2017). For Spanish ASR, we used another dedicated G2P tool (Tejedor-García 2021) based on SAMPA (Speech Assessment Methods Phonetic Alphabet) (Wells et al. 1997). The pronunciation model or lexicon for Flemish ASR is based on Fonilex (Mertens and Vercammen 1997), which provides multiple Southern

Dutch phonemic transcriptions for 170k common Dutch words. For Dutch ASR the pronunciation lexicons were acquired from the CGN lexicon. The CGN lexicon comprises almost all types (unique word forms) that occur in the CGN corpus. The CGN lexicons provide pronunciation for 181k Dutch words. For Irish ASR with Wav2Vec2, we did not use any language model. For all out-of-vocabulary words (OOV words), the lexicon that is used during speech decoding is supplemented with automatically generated phonetic transcriptions obtained via these G2P tools.

3.3 Experimental setting

3.3.1 Kaldi ASR

For English, Spanish, Dutch, and Flemish ASRs a deep neural network (DNN) based large vocabulary continuous speech recognition (LVCSR) system for these languages was developed using the open-source Kaldi speech recognition toolkit (Povey et al. 2011). A standard Kaldi speech recognition model consists of an acoustic model, pronunciation lexicons, and an n-gram language model. For the acoustic model, a time-delayed neural network (TDNN) with the front end of the Convolutional Neural Network (CNN) model was trained on the training datasets (Peddinti et al. 2015). The acoustic model training is done in the Kaldi toolkit using the NNET3 library for CNN-TDNN. In all English, Spanish, Dutch and Flemish ASR the same Kaldi Librispeech recipe[2] is used. Before the training of the DNNs, the data were initially aligned using the triphone GMM-HMM and trained using Mel Frequency Cepstral Coefficient (MFCC) features. Apart from that, linear discriminative analysis (LDA), maximum likelihood linear transformation (MLLT), feature space maximum likelihood linear regression (fMLLR), and speaker adaptive training (SAT) were also applied to them beforehand. The 40-dimensional high-resolution MFCC feature vector was stacked with 100-dimensional extracted i-vectors.
In the Kaldi speech recognition toolkit, there are two commonly used data augmentation techniques for speech data i.e. speed perturbation (Ko et al. 2015) and spectral augmentation (SpecAugment) (Park et al. 2019). Both on-the-fly data augmentation techniques work during the training improving the flexibility of training and helping in reducing the required disk space. The speed perturbation technique created three copies of training data with the speed warping factor of 0.9, 1.0, and 1.1. By using the SpecAugment tool, the time and frequency domain bands of log mel-spectrogram of the single training speech data are randomly masked up to certain extents. These methods are known to yield an improvement in ASR scores, see e.g. (Rath et al. 2013).
The Time Delayed Neural Network (TDNN) model is trained with the Lattice-Free Maximal Mutual Information (LF-MMI) criterion (Povey et al. 2016). This MMI function basically increases the probability of the reference transcript of the particular audio file while decreasing the probability of all other transcripts. Currently, the lattice-free MMI (i.e. LF-MMI) method (Hadian et al. 2018) achieves state-of-the-art results on many speech recognition tasks (Povey et al. 2016) (Peddinti et al. 2018). This method, like CTC, uses a sentence level posterior for training the neural network but unlike end-to-end approaches, still, loosely relies on alignments from a pretrained HMM-GMM model. In this TDNN training, a regularization technique ('leaky HMM') is used to deal with overfitting due to sequential training. A 'leaky HMM' allows the transition probability from each stage to every other stage, to ensure gradual forgetting of context. It's equivalent to stopping and restarting the HMM with some probability on each frame. The *leaky-hmm-factor* is set to 0.1 (Povey et al. 2016). The dropout is different at various points of training. From the beginning of training to 20% of training the dropout ratio is 0 and increases linearly to 50% at the halfway of training and again decreases to 0 linearly at the end of training.

In the testing stage of Kaldi ASR, the weighted Finite-state Transducer (WFST) based lattice generation method is applied in the actual speech decoder. The resulting weighted FST 'WFST' is

2. https://github.com/kaldi-asr/kaldi/tree/master/egs/librispeech/s5

the compose of four underlying finite state transducers:

$$S \equiv HCLG = \min(\det(H \circ C \circ L \circ G))$$

where H represents the Hidden Markov Model (HMM) structure, C denotes the FST with phonetic context-dependency information, L is the FST containing the lexicon, G denotes the FST containing the grammar, and $\circ$ represents the 'composition' operation of WFSTs. On an arc in HCLG, the input label is the identifier of clustered context-dependent HMM state, the output label corresponds to a word, and the weight typically represents a negated log probability. Beam pruning is used to keep the search graph (or lattice) tractable.

3.3.2 Wav2Vec2

An end2end model, Wav2Vec2.0 (Baevski et al. 2020), is used to extract speech representations from raw audio files in a self-supervised learning scenario and use these representations for ASR-specific tasks. In the SignOn project, Wav2vec2 is applied for Irish as it is claimed to can achieve state-of-the-art results when first trained on a large amount of unlabeled speech data and then finetuned on a small-size specific labelled data (e,g., as small as 10 minutes). This scenario is ideal for Irish Speech Recognition as large Irish speech corpora along with the reference transcriptions are not yet available.

In Wav2vec2.0's two-stage training approach, the first stage (called 'pretraining') typically uses a huge amount (hundred thousands of hours of speech, from different languages) of unlabeled data. By Wav2vec2.0, the spoken audio is encoded using a multilayer neural network. In order to create a contextualized representation of the speech audio, masking is applied to the resulting latent speech representation after encoding. The contrastive loss on which the model is trained is computed using Gumbel Softmax. It is an efficient gradient estimator that replaces non-differentiable samples from the categorical distribution with differentiable samples from the new Gumbel softmax distribution. This distribution has the intrinsic effect that it can be smoothly annealed to a categorical distribution (Jang et al. 2016). The second stage (called 'finetuning') is task-dependent. During finetuning, the network's connection weights from the prefinal to the final layer are adjusted using labelled data using Connectionist Temporal Classification (CTC) loss. Fine-tuning is typically carried out on the basis of a small amount of labeled data. As a consequence, by using a generally available pretrained model, we can obtain an Irish ASR by finetuning such a pretrained model by using a small amount of labeled Irish data, despite the lack of large Irish speech corpora.

In contrast with the modular knowledge-based approach of Kaldi, Wav2Vec2 is a data-oriented, end2end approach for Automatic Speech Recognition. Wav2vec2 was released in September 2020 soon after the superior performance of Wav2Vec2 was demonstrated on the Librispeech corpus. Facebook AI demonstrated the multilingual version of Wav2Vec2 named XLSR, this acronym referring to 'cross-lingual speech representations' because it has been pretrained on 53 languages. Its successor XLS-R (Conneau et al. 2020) (Babu et al. 2021) uses a network with 300 million to two billion parameters and is pretrained on over half a million hours of audio data selected from 128 languages. Both XLSR and XLS-R pre-trained models have been exposed to Common Voice Irish. In our experiments, all pre-trained models were finetuned for 30 epochs using a concatenation of the training data of the Common voice Irish and Living audio data as mentioned in section 3.1.1 (4th point). Identical training hyperparameters values were used for all fine-tuning.

3.4 Testing

For a typical Large Vocabulary Continuous Speech Recognition (LVCSR) system, a test set should be large enough to differentiate between different recognizers and recognizer settings. Statistical arguments play a major role here. Usually, two hours of speech (about 20000 words) are sufficient to statistically differentiate between ASR systems with a difference of 0.3% WER at 20% WER.

Table 2: Word Error Rate (%) of benchmarking test set for all SignON languages

Test Set	Hours	Word Error Rate (WER)
English		
GigaSpeech Dev	12.5	23.69%
GigaSpeech Test	40	22.67%
LibriSpeech Test Clean	5.4	13.71%
LibriSpeech Test Others	5.1	33.87%
Spanish		
Common Voice Test	26.1	15.69%
Common Voice Dev	25.9	13.68%
Irish		
Common Voice Irish + Living Audio	0.55	25.94%
Flemish		
CGN Dev	15.5	14.02%
N Best 2008 Evaluation	2	10.99%
Dutch		
CGN Dev_S	0.5	5.85%
CGN Dev_T	1.97	28.60%
Common-Voice Test (Tagged with NL Dutch accent)	4.4	24.78%

In SignON, we used different benchmark test sets for each language to estimate the results of each ASR. In section 4 we present the results in terms of WER for each language.

4. Results

How well the acoustic model (AM) and language model (LM) training data match the test conditions continues to be a key factor in ASR accuracy. Therefore, being able to evaluate an ASR system's accuracy in a certain target environment is crucial. The standard measure for assessing the effectiveness of a large vocabulary continuous speech recognition (LVCSR) system is the word error rate (WER). The ASR system's predicted word sequence is compared to a reference transcription, and the total number of errors is determined by adding the sum of the substitutions (S), insertions (I), and deletions (D). For N number of total words in reference transcription, the Word Error Rate (WER) is defined as

$$WER = \frac{I + D + S}{N} * 100$$

Table 2 provides an overview of the results. We tested our trained ASRs on the test sets of particular training datasets as well as different benchmark datasets to determine how the ASR performs in the same environment and a different environment than it was trained. The overview of the test sets is briefly described in section 3.1.1.

5. Discussion and conclusion

In this work, we presented our progress in the ASR component of the SignON application that is built in the European SignON project (Grant Agreement No 101017255). We discussed the technical and design details about how ASR is communicating with the SignON app through a REST API with the

AudioServer (AS), how the Audioserver works as webservice, and the various issues that are to be addressed concerning privacy, latency, and communication protocols. The current ASR component is external to the SignON application and communicates with the SignON app via the internet. It is a highly scalable and flexible component, allowing the extension of many more languages and acoustic and language models. Scalability in combination with low latency and high performance is expected to be a big asset for the sustainability and end user-friendliness of the SignON application.

In the SignON project, the ASR component will not only transcribe the input audio files for multiple languages; the transcriptions will also be used for the NLP and machine translation along the downstream pipeline. An important issue to address is that the current training and testing procedures use audio and text material that is specified in terms of context, language use, and acoustical characteristics that may largely deviate from what we can expect from the SignON app's input. The SignON app primarily focuses on hard-of-hearing persons and persons with a cochlear implant. The future work of ASR in terms of the SignON project is therefore directed to improve our existing recognizers for deaf/hard-of-hearing (DHH) speakers. To that end, we have defined a specific use case (with a focus on the topic of hospitality). We will record the speech from DHH people on specified topics. With end-to-end ASR models, a trade-off between latency and WER also comes into place. Currently, our ASR web service is working with Kaldi ASR and it would be a challenge (also in terms of hardware) to integrate both end-to-end Wav2Vec2 ASR and Kaldi ASR into a single web service. Future work will also include the improvement of the performance of the existing recognizers in terms of WER and robustness.

Above we already addressed the difference between training and test conditions. Since the actual SignON app is not in full use yet, it remains to be seen how the ASR component and the numerous other components function in real-life situations in conditions that may be adverse. Suppose test material deviates from training material in terms of acoustics, speakers, accents, background noise et cetera. In that case, the ASR performance may gradually drop compared to the results shown in the table above. However, to what extent this deterioration really harms the overall applicability of the SignON app is to be seen, since the WER of the ASR component itself is only a weak indicator of the success of spoken dialogue systems. The NLP component, generating the semantic frames, is equally important. The use of a flexible ASR in an application such as SignON is likely to boost interactive cross-lingual communication by reducing translation costs while being supported by the rapid growth in capabilities of speech-controlled interfaces.

As mentioned above, we aim to use a specific case to improve the ASR component in the context of the SignOn project. A particularly interesting and important user group in SignON is formed by users who are deaf or hard of hearing (DHH). For this group, the usability of speech-controlled devices may be limited due to high error rates (which may be about 5 times higher compared to regular hearing users). Globally, 1.5 billion people live with some degree of hearing loss and this number could rise to over 2.5 billion by 2050 (Organization et al. 2018), and around 430 million people require rehabilitation services for their hearing loss (Thammaiah et al. 2017). In SignON we know that significant advances in speech recognition software are required for deaf use of speech-controlled interfaces, and a roadmap for this specific use case is being developed.

6. Acknowledgement

This work has been conducted within the SignON project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017255.

References

Ardila, Rosana, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber (2020), Common voice: A massively-multilingual speech corpus, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, European Language Resources Association, Marseille, France, pp. 4218–4222. https://aclanthology.org/2020.lrec-1.520.

Babu, Arun, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, et al. (2021), Xls-r: Self-supervised cross-lingual speech representation learning at scale, *arXiv preprint arXiv:2111.09296*.

Baevski, Alexei, Henry Zhou, Abdelrahman Mohamed, and Michael Auli (2020), wav2vec 2.0: A framework for self-supervised learning of speech representations.

Braude, David A, Matthew P Aylett, Caoimhín Laoide-Kemp, Simone Ashby, Kristen M Scott, Brian Ó Raghallaigh, Anna Braudo, Alex Brouwer, and Adriana Stan (2019), All together now: The living audio dataset., *INTERSPEECH*, pp. 1521–1525.

Cardellino, Cristian (2019), Spanish Billion Words Corpus and Embeddings. https://crscardellino.github.io/SBWCE/.

Chelba, Ciprian, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson (2013), One billion word benchmark for measuring progress in statistical language modeling, *Technical report*, Google. http://arxiv.org/abs/1312.3005.

Chen, Guoguo, Shuzhou Chai, Guanbo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, Mingjie Jin, Sanjeev Khudanpur, Shinji Watanabe, Shuaijiang Zhao, Wei Zou, Xiangang Li, Xuchen Yao, Yongqing Wang, Yujun Wang, Zhao You, and Zhiyong Yan (2021), Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio.

Conneau, Alexis, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli (2020), Unsupervised cross-lingual representation learning for speech recognition, *arXiv preprint arXiv:2006.13979*.

Demuynck, Kris, Antti Puurula, Dirk Van Compernolle, and Patrick Wambacq (2009), The esat 2008 system for n-best dutch speech recognition benchmark, *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*, IEEE, pp. 339–344.

Hadian, Hossein, Hossein Sameti, Daniel Povey, and Sanjeev Khudanpur (2018), End-to-end speech recognition using lattice-free mmi, *INTERSPEECH*.

Jang, Eric, Shixiang Gu, and Ben Poole (2016), Categorical reparameterization with gumbel-softmax. https://arxiv.org/abs/1611.01144.

Kaiser, L (2017), Accelerating deep learning research with the tensor2tensor library, *URL: https://ai.googleblog.com/2017/06/accelerating-deep-learning-research.html*.

Kessens, Judith and David A van van Leeuwen (2007), N-best: The northern-and southern-dutch benchmark evaluation of speech recognition technology, *Eighth Annual Conference of the International Speech Communication Association*, Citeseer.

Ko, Tom, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur (2015), Audio augmentation for speech recognition, *INTERSPEECH*.

Mertens, Piet and Filip Vercammen (1997), Fonilex manual, Leuven.

Oostdijk, Nelleke (2000), The spoken dutch corpus project, *The ELRA newsletter* **5** (2), pp. 4–8.

Ordelman, Roeland J.F., Franciska M.G. de Jong, Adrianus J. van Hessen, and G.H.W. Hondorp (2007), Twnc: a multifaceted dutch news corpus, *ELRA Newsletter*.

Organization, World Health et al. (2018), Addressing the rising prevalence of hearing loss, World Health Organization.

Panayotov, Vassil, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur (2015), Librispeech: an asr corpus based on public domain audio books, *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, IEEE, pp. 5206–5210.

Park, Daniel S., William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le (2019), SpecAugment: A simple data augmentation method for automatic speech recognition, *Interspeech 2019*, ISCA. https://doi.org/10.21437

Peddinti, Vijayaditya, Daniel Povey, and Sanjeev Khudanpur (2015), A time delay neural network architecture for efficient modeling of long temporal contexts, *Sixteenth annual conference of the international speech communication association.*

Peddinti, Vijayaditya, Yiming Wang, Daniel Povey, and Sanjeev Khudanpur (2018), Low latency acoustic modeling using temporal convolution and lstms, *IEEE Signal Processing Letters* **25** (3), pp. 373–377.

Povey, Daniel, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. (2011), The kaldi speech recognition toolkit, *IEEE 2011 workshop on automatic speech recognition and understanding*, number CONF, IEEE Signal Processing Society.

Povey, Daniel, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur (2016), Purely sequence-trained neural networks for asr based on lattice-free mmi., *Interspeech*, pp. 2751–2755.

Rath, P. Shakti, Daniel Povey, Karel Veselý, and Jan Černocký (2013), Improved feature processing for deep neural networks, *Proceedings of Interspeech 2013*, number 8, International Speech Communication Association, pp. 109–113. https://www.fit.vut.cz/research/publication/10432.

Stolcke, A. (2000), Entropy-based pruning of backoff language models, arXiv. https://arxiv.org/abs/cs/0006025.

Stolcke, Andreas (2002), Srilm-an extensible language modeling toolkit, *Seventh international conference on spoken language processing.*

Tejedor-García, Cristian (2021), Cristiantg/g2p_spanish: A grapheme to phoneme (g2p) tool for spanish., *GitHub.* https://github.com/cristiantg/g2p_spanish.

Tejedor-García, Cristian (2022), Cristiantg/audioserver, *GitHub.* https://github.com/cristiantg/audioserver.

Thammaiah, Spoorthi, Vinaya Manchaiah, Vijayalakshmi Easwar, Rajalakshmi Krishna, and Bradley McPherson (2017), Community-based hearing rehabilitation: Implementation and outcome evaluation, *Perspectives of the ASHA Special Interest Groups* **2** (17), pp. 83–95, ASHA.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017), Attention is all you need, *Advances in neural information processing systems.*

Voxforge.org (n.d.), Free speech... recognition (linux, windows and mac) - voxforge.org. accessed 06/25/2014.

Wells, John C et al. (1997), Sampa computer readable phonetic alphabet, *Handbook of standards and resources for spoken language systems* **4**, pp. 684–732, Mouton de Gruyter Berlin.

7. Appendices

7.1 Appendix 1

The following text shows the code at the client side to make the connection to the Audioserver. Its basic functionality consists in three steps: (1) login, (2) download/upload file or link and retrieve audio ticket, and (3) decoding and downloading results, in a specific (CTM) format. The CTM formatted output consists of transcribed words, the confidence score for the specific word and the duration at which a particular word was spoken.

```
username = "****"
password = "****"
# Option A: Download a file
urlfile = "https://cls.ru.nl/~ltenbosch/DitIsEenTest.wav"
# Option B: Upload a file
# Absolute path
file_path = '/vol/tensusers4/aparikh/SignON/sample.wav'

# 1. LOGIN
r = requests.post('https://restasr.cls.ru.nl/auth/login', json={
"username": username,
"password": password
}, headers = {'Content-Type': 'application/json'}
)
print(f"Status Code: {r.status_code}, Response: {r.json()}")
token = r.json()['data']['access_token']


# 2. DOWNLOAD an audio file
r2 = requests.post('https://restasr.cls.ru.nl/users/'+username+'/audio/download',
    json={
"urlfile": urlfile},
headers = {"Authorization": "Bearer " + token}
)
print(f"Status Code: {r2.status_code}, Response: {r2.json()}")


# 2.bis UPLOAD an audio file

file_filename = os.path.basename(file_path)
multipart_form_data = {'file':(file_filename,open(file_path, 'rb'),'audio/wav')}
headers = { 'Authorization': 'Bearer ' + token }

r2 = requests.post('https://restasr.cls.ru.nl/users/'+username+'/audio',headers=
    headers,
files=multipart_form_data)
print(f"Status Code: {r2.status_code}, Response: {r2.json()}")


# 3. DECODE
id_audio = r2.json()['data']['filename']
params = {"code": 1,
"text": "custom text",
"keep": True,
"beam": 18,
}
```

```
r3 = requests.post('https://restasr.cls.ru.nl/users/'+username+'/audio/'+id_audio,
json=params, headers = {'Authorization': "Bearer "+token})

print(f"Status Code: {r3.status_code}, Response: {r3.json()}")
text = r3.json()['data']['nbest']
```

Listing 1: Client side Connection with Audioserver

```
HELLO NICE TO ME TO
0: {'channel': 1, 'tbeg': 0, 'dur': 0.81, 'word': 'HELLO', 'score': 0.89}
1: {'channel': 1, 'tbeg': 0.95, 'dur': 0.76, 'word': 'NICE', 'score': 1}
2: {'channel': 1, 'tbeg': 1.71, 'dur': 0.15, 'word': 'TO', 'score': 1}
3: {'channel': 1, 'tbeg': 1.86, 'dur': 0.09, 'word': 'ME', 'score': 0.73}
4: {'channel': 1, 'tbeg': 1.95, 'dur': 0.06, 'word': 'TO', 'score': 0.37}
```

Listing 2: Output from Audioserver: Time marked Conversations (CTM)