# A comparison of cloud-based speech recognition engines

**Andrey L. Herchonvicz[1], Cristiano R. Franco[1], Marcio G. Jasinski[1]**

[1]Senior Sistemas S/A
Departamento de Pesquisa Aplicada
Blumenau – SC – Brasil.

{andrey.lucas,cristiano.franco,marcio.jasinski}@senior.com.br

***Abstract.*** *Human-machine interaction is present in our routines and has become increasingly natural these days. Devices can record a person's speech, transcribe into text and execute tasks accordingly. This kind of interaction provides more productivity for several operations since it allows users to have hands free through a more natural interface. Moreover, the speech recognition engines need to assure reliability and speed. However, the maturity of speech recognition systems vary from providers and most importantly accordingly to the language. For instance, Brazilian Portuguese language has a particularity of using several foreign terms, especially if we consider corporate environments. In this paper, an experiment was conducted, to evaluate three speech recognition engines regarding accuracy and performance: Bing Speech API, Google Cloud Speech and IBM Watson Speech to Text. To obtain the accuracy value, we used a well-known string similarity algorithm. The results showed a high level of accuracy for Google Cloud Speech and Bing Speech API. However, the best accuracy provided by Google services came with a cost on performance – requiring additional time to provide the speech to text transcription.*

## 1. Introduction

Automatic Speech Recognition (ASR) for dictation, voice commands and search by voice has become an essential feature of modern applications running on mobile smartphone and wearable devices. Such evolution is driven by a significant improvement in speech recognition accuracy. According to the authors [Schalkwyk et al. 2010, Xu et al. 2015], several factors have contributed to this improvement. First, the constant increase of computing power, allowing a mobile device to have as much processing power as a regular PC. Second, the popularization of voice services like Google Voice Search, Siri, and Cortana, allowed rich data gathering to train and improve acoustic models. Finally, the algorithms and techniques used for speech recognition have evolved considerably in the last couple of years [Picheny 2015, Sui et al. 2015, Obin et al. 2014].

As reported by Hearst [Hearst 2011], the ASR provides a vital role on natural interfaces, where users interact more naturally with the system. Smartphone usage also enforces ASR systems usage since mobile devices provide less productive interfaces. A feature like that is even more critical for users who tend to use the smartphone while driving. In agreement with to Klauer [Klauer et al. 2014], such behavior increases the risk of a crash by a factor of 4. Providing ASR capability in software applications is not only a better way to interact with the system but also a much more safe one.

While Speech Recognition Systems (SRS) has been improved for practical usage, some challenges remain. For instance, the multi-language speech recognition is still not

solved due to the sparseness of speech and text data with corresponding pronunciation dictionaries, the lack of language conventions, and the gap between technology and language expertise [Schultz et al. 2013]. As a consequence of globalization, the English language is often used by non-English cultures through the adoption of words and expressions. The Brazilian corporate vocabulary is no exception on this matter since executives tend to mix words from Portuguese and English in a single phrase. Thanks to the rich statistic data available on cloud-based services, some SRS are capable of handling multi-language at a certain level. If a user dictates a sentence in English and then dictates the second sentence in Portuguese, the recognition usually occurs on both sentences, as long as both languages are supported. However, mixing both languages in the same sentence will mostly result in a speech recognition error. The same issue also occurs with names, acronyms and specific corporate terms [Sak et al. 2013, Stemmer et al. 2001].

A whole different problem for the ASRs implies the difficulty of isolating only one person's speech in an environment with external noises or several people speaking. This phenomenon is called The Cocktail Party Effect [Arons 1992]. According to the authors [McGraw et al. 2016, Amodei et al. 2015], companies like Google and Baidu are investing in the area of speech recognition using different approaches from traditional ones to deep neural networks. These studies show a significant improvement in accuracy, including for different accents in a noisy environment.

This paper introduces a comparison of speech recognition tools, to evaluate the accuracy and performance. The main contributions of this work are:

- Develop an experiment to evaluate ASR, cloud-based engines, Bing Speech API, Google Cloud Speech API, and IBM Watson Speech to Text;
- Evaluate the accuracy of ASR tools, using different string similarity approaches;
- Compare response time between the tools.

The rest of this paper is divided as follows. In Section 2, we describe important works that tie with ours. An overview of how ASR systems work in discussed in Section 3. In Section 4 we present the proposed method, the string similarity algorithms and how they operate. Results and discussions are demonstrated in Section 5. Finally, in Section 6 we present our conclusion.

## 2. Related Work

Regarding the challenges of foreign words, expressions, acronyms and names for the SRS, Škraba [Škraba et al. 2014, Škraba et al. 2015] and Shen [Shen et al. 2015], demonstrates how the pronunciation of words by a non-native speaker causes recognition errors. Aryal and Gutierrez-Osuna [Aryal and Gutierrez-Osuna 2014] introduce a technique to reduce accent in voice conversion for non-native speakers. Finally, Aleksic [Aleksic et al. 2015] presents the challenges of recognizing contact names in voice commands, because of its low prior probability.

A study conducted by Gaida [Gaida et al. 2014] compared the open-source speech recognition tools HTK, CMU Sphinx and Kaldi for Brazilian Portuguese language. Their approach compared the results of transcribed text using word error rate (WER) metric, proposed by Klakow and Peters [Klakow and Peters 2002], instead of string similarity metrics.

Following, Lange and Suendermann-Oeft [Lange and Suendermann-Oeft 2014] tuned the CMU Sphinx with 258K utterances of recorded speech with the objective of outperforming Google Cloud Speech API. Nevertheless, Google showed a result significantly higher than CMU Sphinx.

To measure ASR engines, Blanchard [Blanchard et al. 2015], evaluated Google Cloud Speech API and Bing Speech API, among others. In this work, we do not include AT&T Watson since it was replaced by IBM Watson Speech to Text service. Similarly, the Microsoft Speech SDK 5.1 was deprecated to enforce Bing Speech API usage. Finally, we discarded CMU Sphinx since it is not a cloud available service and it has shown on the literature that its accuracy might be suitable to the local application (without internet connection), but it cannot overcome cloud engines without considerable efforts.

## 3. Speech Recognition

Transcribing the human voice, it is not an easy task. Even if two individuals speak the same sentence, the properties of both acoustic wave generated are entirely different from each other. Another problem is how to quickly find a match of chunks of sound in a vast database. Fortunately, SRS store sets of properties that represent the words, known as models.

Traditionally, ASR systems are divided into front-end and back-end. The front-end split the input audio on chunks and then extract a numeric representation, called feature vector. On the other hand, the back-end is responsible for searching for a feature vector match. The decoder manages the search through the linguistic database [AbuZeina et al. 2012].

A linguistic database usually is formed by three parts: a lexicon model, which is also known by a phonetic dictionary, an acoustic model and a language model. According to authors [Eulitz and Lahiri 2004], a lexicon model contains a map of all words that can be recognized and how the words are spoken through phonemes. This model can store the same word using the different type of phonemes.

Every word is formed by a set of distinct sounds known as phonemes or phonetic features, which can be represented statistically and stored in the acoustic model. The language model is responsible for delimiting the search for the next word based on the previously recognized words, according to Jelinek [Jelinek et al. 1991], which provides performance and improves accuracy. The most common language models are based on grammars and statistics. The grammar-based language models contain exact rules about what can be spoken and in what order. On the other hand, with statistical language models, every combination of words is possible, but the probability of this sequence can vary.

## 4. String comparison methods

To compare between the expected sentence $e$ and the text transcribed by the engines $t$, we used three methods that calculate a similarity of strings. Next, we calculate an average value $SC$ of the string similarity algorithms. The standard aims to reduce the impact of the weakness of every string similarity comparison approach. The average value is composed by Levenshtein Distance $L$, Sørensen-Dice Coefficient $SD$ and Jaro-Winkler Distance $JW$ string similarity algorithms. These algorithms were chosen because they are well

known and established, according to authors [Zou et al. 2004, Choudhury et al. 2007]. The equation 1 shows how the similarity coefficient obtained from the speech recognition sentence and expected sentence.

$$SC(t,e) = \frac{L(t,e) + SD(t,e) + JW(t,e)}{3} \tag{1}$$

### 4.1. Levenshtein Distance

Also called Edit Distance, the Levenshtein Distance algorithm calculates the number of operations necessary to transform one string into another. The number of operations obtained by the sum of insertions, deletions, and substitutions [Gilleland et al. 2009]. The Equation 2 presents the algorithm. Equation 3 presents how to obtain the similarity of the sentences, where $l$ is Levenshtein Distance, and $x$ and $y$ are the lengths of the sentences.

$$l_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} l_{a,b}(i-1,j) + 1 \\ l_{a,b}(i,j-1) + 1 \\ l_{a,b}(i-1,j-1) + 1_{a_i \neq b_j} \end{cases} & \text{otherwise.} \end{cases} \tag{2}$$

$$C = 1 - \left( \frac{l}{max(|x|,|y|)} \right) \tag{3}$$

### 4.2. Sørensen-Dice Coefficient

A bi-gram is a sequence of two characters in a string. Sørensen-Dice coefficient calculates the division between the number of common bi-grams in the input strings multiplied by 2, with the sum of the quantity of the bi-grams [Javadi-Moghaddam and Kollias 2014]. Given that $n_x$ and $n_y$ are the numbers of bi-grams in the strings $x$ and $y$, and $n_t$ is the number of intersection between those bi-grams, the coefficient obtained by the equation 4.

$$Sim(x,y) = \frac{2n_t}{n_x + n_y} \tag{4}$$

### 4.3. Jaro-Winkler Distance

The Jaro-Winkler Distance's algorithm is divided into two parts. The first one is the Jaro implementation itself, that considers the matched characters between two strings, regardless of the order. If the characters matched in a different order, it uses a window of tolerance for the number of transpositions [Gueddah et al. 2015]. Jaro Distance is calculated by equation 5:

$$\theta = \begin{cases} 0 & \text{if } c = 0 \\ W_1 \cdot \frac{c}{d} + W_2 \cdot \frac{c}{r} + W_t \cdot \frac{c-\tau}{c} & \text{otherwise} \end{cases} \tag{5}$$

Where:

$W_1$, $W_2$ and $W_t$ - Weight. In this case, if we want a value between 0 and 1, we apply the weight of $\frac{1}{3}$;

$d$ and $r$ - Length of the first and second input strings, respectively;

$c$ - Quantity of match characters in the same order;

$\tau$ - Quantity of transpositions limited to the window of tolerance $\lfloor \frac{max(d,r)}{2} \rfloor - 1$;

The second part of the algorithm is the Winkler implementation, where Jaro-Winkler distance can be obtained using the value of $\theta$ which is described by the equation 6.

$$\theta = \theta + i \cdot 0.1 \cdot (1 - \theta) \tag{6}$$

## 5. Experimental results

In order to evaluate ASR tools, an experiment with 15 men and 15 women Brazilian native speakers were conducted. Every participant was lead to a quiet environment and then requested to complete a set of actions in the application using only voice commands. The experiment consisted of a script with 20 sentences containing common terms in our corporate environment, such as foreign words, names, and acronyms. Table 1 shows the sentences identified by an ID.

Figure 1 shows the architecture of the application developed to conduct the experiment. The solution is designed to record every participant sentence and also evaluate ASR engines. The experiment is controlled by a mobile app developed on Android platform. This app shows the sequence of voice commands to the participant and then gather the participant speech. The voice is recorded in PCM format with one channel (mono), audio depth of 16 bits per sample and sample rate of 16000Hz. Once the audio is captured, the application sends the data to a back-end service.

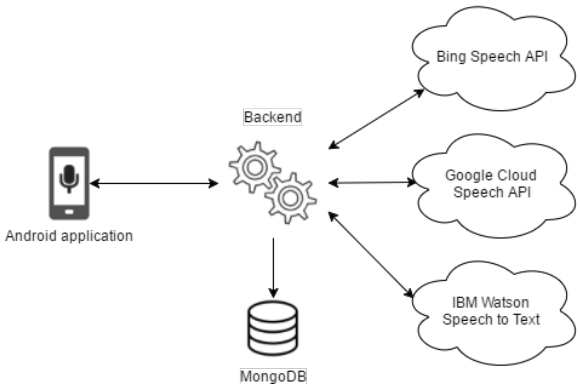


**Figure 1. Architecture of the experiment**

Once the back-end receives the request with the audio data, it stores it into a MongoDB. Next, the same audio data, without normalization, is sent to all speech recognition engines and the time begins to count. After receiving the response from the engines, the back-end registers the request/response elapsed time. Finally, the similarity coefficient ($SC$) is calculated between the expected sentence and the transcribed text.

**Table 1. Sentences used for the experiment**

| ID | Sentence |
|---|---|
| 1 | PDI da minha equipe<br>*PDI of my team* |
| 2 | Exibir meus feedbacks<br>*Show my feedbacks* |
| 3 | Evolução salarial da equipe<br>*Salary evolution of the team* |
| 4 | Perfil público de Luan José Pereira<br>*Public profile of Luan José Pereira* |
| 5 | Consultar depósitos do FGTS<br>*Check FGTS deposits* |
| 6 | Calcular taxa de turnover<br>*Calculate turnover rate* |
| 7 | Comparar salários dos meus liderados<br>*Compare salary of my employees* |
| 8 | Evolução salarial de Caio Gonçalves Lima<br>*Salary evolution of Caio Gonçalves Lima* |
| 9 | Gerar SEFIP normal<br>*Generate normal SEFIP* |
| 10 | Exibir matriz Nine Box<br>*Show Nine Box matrix* |
| 11 | Contrato de resultado<br>*Contract of result* |
| 12 | Ficha registro de Julieta Santos Ribeiro<br>*Registration form of Julieta Santos Ribeiro* |
| 13 | Contribuintes do ICMS das unidades federadas<br>*ICMS taxpayers of federated units* |
| 14 | Cadastrar contrato de Royalty<br>*Register of Royalty contract* |
| 15 | Imprimir meu holerite<br>*Print my payslip* |
| 16 | Transmitir arquivos para o eSocial<br>*Send files to eSocial* |
| 17 | Listar processo do Workflow<br>*List workflow process* |
| 18 | Detalhar rubricas da folha de pagamento<br>*Detail payroll items* |
| 19 | Consultar valor de IRRF<br>*Check IRRF value* |
| 20 | Minha programação de férias<br>*My vacation schedule* |

Figure 2 shows the linear variation of $SC$ value from every sentence and every ASR engine, where 1 means that the API has fully recognized a sentence for all the participants and 0 means a complete failure.

The evaluation of total values indicates that the engine of Google obtained the best results with an average accuracy of almost 0.978, followed close by Bing Speech API with 0.945. The average value of IBM Watson Speech to Text was close to 0.881. Observing the results by sentence, API of Microsoft recognized correctly the phrases represented by IDs 3 and 20. These phrases were transcribed correctly for all participants. Google Cloud Speech API obtained the maximum accuracy score on the sentences 8, 12 and 20, while the best score of IBM Watson Speech to Text was 0.998 in sentence 3.

The sentence 17 obtained the worst accuracy by Microsoft and Google, with a score about 0.781 and 0.897, respectively. This is justified by the difficulty of recognizing correctly the word “workflow”, which is a foreign word for the Brazilian Portuguese language. The sentence number 2 got the lowest score by the solution of IBM, with a value around 0.635. In this case, the reason was the word ”feedbacks,” another foreign

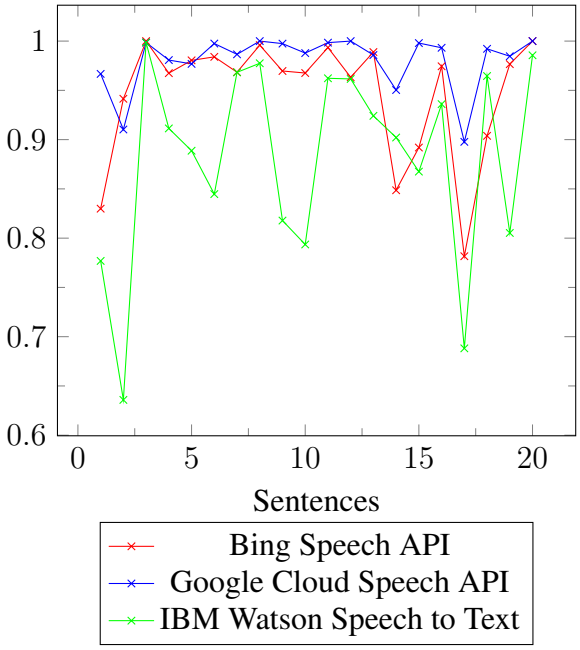


**Figure 2. Linear variation by API**

term.

The evaluation of correct answers by each speech recognition engine is shown on Table 2. Google Cloud Speech API has a clear advantage, with a score of 487 correct sentences out of 600, which represents almost 81%. The second best results were from Bing Speech API with 376 of 600 or close to 62%. The engine of IBM appears in the last place, with 186 of 600 right answers, representing around 31%.

**Table 2. Right answers by sentence**

| ID | Bing Speech API | Google Cloud Speech API | IBM Watson STT |
|---|---|---|---|
| 1 | 5 | 24 | 0 |
| 2 | 0 | 5 | 0 |
| 3 | 30 | 29 | 29 |
| 4 | 18 | 24 | 0 |
| 5 | 25 | 22 | 2 |
| 6 | 27 | 29 | 0 |
| 7 | 18 | 25 | 17 |
| 8 | 29 | 30 | 21 |
| 9 | 22 | 29 | 0 |
| 10 | 25 | 27 | 0 |
| 11 | 26 | 28 | 17 |
| 12 | 19 | 30 | 20 |
| 13 | 26 | 24 | 13 |
| 14 | 1 | 15 | 1 |
| 15 | 13 | 29 | 16 |
| 16 | 23 | 28 | 0 |
| 17 | 7 | 8 | 0 |
| 18 | 9 | 28 | 23 |
| 19 | 23 | 23 | 0 |
| 20 | 30 | 30 | 27 |
| Total | 376/600 | 487/600 | 186/600 |

The critical factors for the sentences where recognition errors occurred were the

existence of words in Portuguese and English language, acronyms, names, and specific corporate terms. While Google Cloud Speech API recognized all of these factors at least one time and Bing Speech API failed to recognize only the foreign word “feedbacks”, IBM Watson Speech to Text failed to recognize “feedbacks”, “turnover”, “Nine Box”, “Workflow”, “PDI”, “IRRF”, the name “Luan” and finally the specific corporate terms “SEFIP” and “eSocial”. Furthermore, the results showed no significant difference between men and women.

Nevertheless, the higher accuracy came with a price for Google, taking an average time of 7170.3 milliseconds by sentence. IBM’s average time was 3024.7 milliseconds, while the fastest was the engine of Microsoft with an average of 2659.3 milliseconds. In other words, Google Cloud Speech API was around 4.5 seconds or 269% and 4 seconds or 237% slower than Bing Speech API and IBM Watson Speech to Text, respectively.

## 6. Conclusion

In the present work, we evaluated the ASR engines Bing Speech API, Google Cloud Speech API and IBM Watson Speech to Text concerning speech recognition accuracy and transcription time. The commands were written in Brazilian Portuguese language, containing foreign words, names, acronyms, and typical corporate terms. In order to compare the ASR engines, we calculated an average value between the expected and the obtained results using string similarity algorithms.

Google Cloud Speech API obtained the best results in accuracy and number of fully recognized sentences, followed by Bing Speech API and IBM Watson Speech to Text. The main factor here was precisely the aspects evaluated by this work. While engines of Google and Microsoft obtained excellent results with foreign words, acronyms, and names, the tool of IBM had difficulty in dealing with them.

Regarding performance for synchronous requests, the engine of Microsoft was the fastest. Right behind came IBM’solution and finally, the slower engine was Google Cloud Speech API. In this scenario, Google obtained a significantly negative result, being almost 4.5 seconds slower than Bing Speech API and IBM Watson Speech to Text, respectively.

Despite the good results, we believe that speech recognition engines can be improved both in accuracy and performance. In the scenarios addressed in this paper, RNN-based solutions have shown promise over traditional HMMs models, not just in accuracy and speed, but also in offline transcription, memory consumption, and battery life, making it ideal for mobile devices.

## References

AbuZeina, D., Al-Muhtaseb, H., and Elshafei, M. (2012). *Cross-Word Arabic Pronunciation Variation Modeling Using Part of Speech Tagging*. INTECH Open Access Publisher.

Aleksic, P., Allauzen, C., Elson, D., Kracun, A., Casado, D. M., and Moreno, P. J. (2015). Improved recognition of contact names in voice commands. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5172–5175.

Amodei, D., Anubhai, R., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Chen, J., Chrzanowski, M., Coates, A., Diamos, G., et al. (2015). Deep speech 2: End-to-end speech recognition in english and mandarin. *arXiv preprint arXiv:1512.02595*.

Arons, B. (1992). A review of the cocktail party effect. *Journal of the American Voice I/O Society*, 12(7):35–50.

Aryal, S. and Gutierrez-Osuna, R. (2014). Can voice conversion be used to reduce non-native accents? In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7879–7883.

Blanchard, N., Brady, M., Olney, A. M., Glaus, M., Sun, X., Nystrand, M., Samei, B., Kelly, S., and D'Mello, S. (2015). A study of automatic speech recognition in noisy classroom environments for automated dialog analysis. In *International Conference on Artificial Intelligence in Education*, pages 23–33. Springer.

Choudhury, M., Thomas, M., Mukherjee, A., Basu, A., and Ganguly, N. (2007). How difficult is it to develop a perfect spell-checker? a cross-linguistic analysis through complex network approach. *arXiv preprint physics/0703198*.

Eulitz, C. and Lahiri, A. (2004). Neurobiological evidence for abstract phonological representations in the mental lexicon during speech recognition. *Journal of cognitive neuroscience*, 16(4):577–583.

Gaida, C., Lange, P., Petrick, R., Proba, P., Malatawy, A., and Suendermann-Oeft, D. (2014). Comparing open-source speech recognition toolkits. *Tech. Rep., DHBW Stuttgart*.

Gilleland, M. et al. (2009). Levenshtein distance, in three flavors. *Merriam Park Software: http://www. merriampark. com/ld. htm*.

Gueddah, H., Yousfi, A., and Belkasmi, M. (2015). The filtered combination of the weighted edit distance and the jaro-winkler distance to improve spellchecking arabic texts. In *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*, pages 1–6.

Hearst, M. A. (2011). 'natural' search user interfaces. *Commun. ACM*, 54(11):60–67.

Javadi-Moghaddam, S.-M. and Kollias, S. (2014). A fuzzy similarity measure for xml documents.

Jelinek, F., Merialdo, B., Roukos, S., and Strauss, M. (1991). A dynamic language model for speech recognition. In *HLT*, volume 91, pages 293–295.

Klakow, D. and Peters, J. (2002). Testing the correlation of word error rate and perplexity. *Speech Communication*, 38(1):19–28.

Klauer, S. G., Guo, F., Simons-Morton, B. G., Ouimet, M. C., Lee, S. E., and Dingus, T. A. (2014). Distracted driving and risk of road crashes among novice and experienced drivers. *New England Journal of Medicine*, 370(1):54–59.

Lange, P. and Suendermann-Oeft, D. (2014). Tuning sphinx to outperform google's speech recognition api. In *Proc. of the ESSV 2014, Conference on Electronic Speech Signal Processing*, pages 1–10.

McGraw, I., Prabhavalkar, R., Alvarez, R., Arenas, M. G., Rao, K., Rybach, D., Alsharif, O., Sak, H., Gruenstein, A., Beaufays, F., et al. (2016). Personalized speech recognition on mobile devices. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 5955–5959. IEEE.

Obin, N., Roebel, A., and Bachman, G. (2014). On automatic voice casting for expressive speech: Speaker recognition vs. speech classification. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 950–954.

Picheny, M. (2015). Ibm watson now brings cognitive speech capabilities to developers. `https://developer.ibm.com/watson/blog/2015/02/09/ibm-watson-now-brings-cognitive-speech-capabilities-developers`. Accessed: 2017-05-21.

Sak, H., Beaufays, F., Nakajima, K., and Allauzen, C. (2013). Language model verbalization for automatic speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8262–8266. IEEE.

Schalkwyk, J., Beeferman, D., Beaufays, F., Byrne, B., Chelba, C., Cohen, M., Kamvar, M., and Strope, B. (2010). *"Your Word is my Command": Google Search by Voice: A Case Study*, pages 61–90. Springer US.

Schultz, T., Vu, N. T., and Schlippe, T. (2013). Globalphone: A multilingual text amp; speech database in 20 languages. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8126–8130.

Shen, H.-p., Wu, C.-h., and Tsai, P.-s. (2015). Model generation of accented speech using model transformation and verification for bilingual speech recognition. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 14(2):6:1–6:24.

Stemmer, G., Nöth, E., and Niemann, H. (2001). Acoustic modeling of foreign words in a german speech recognition system. In *INTERSPEECH*, pages 2745–2748.

Sui, C., Togneri, R., and Bennamoun, M. (2015). Extracting deep bottleneck features for visual speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1518–1522.

Xu, Y., Siohan, O., Simcha, D., Kumar, S., and Liao, H. (2015). Exemplar-based large vocabulary speech recognition using k-nearest neighbors. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5167–5171.

Zou, K. H., Warfield, S. K., Bharatha, A., Tempany, C. M., Kaus, M. R., Haker, S. J., Wells, W. M., Jolesz, F. A., and Kikinis, R. (2004). Statistical validation of image segmentation quality based on a spatial overlap index 1: Scientific reports. *Academic radiology*, 11(2):178–189.

Škraba, A., Koložvari, A., Kofjač, D., and Stojanović, R. (2014). Prototype of speech controlled cloud based wheelchair platform for disabled persons. In *2014 3rd Mediterranean Conference on Embedded Computing (MECO)*, pages 162–165.

Škraba, A., Koložvari, A., Kofjač, D., and Stojanović, R. (2015). Wheelchair maneuvering using leap motion controller and cloud based speech control: Prototype realization. In *2015 4th Mediterranean Conference on Embedded Computing (MECO)*, pages 391–394.