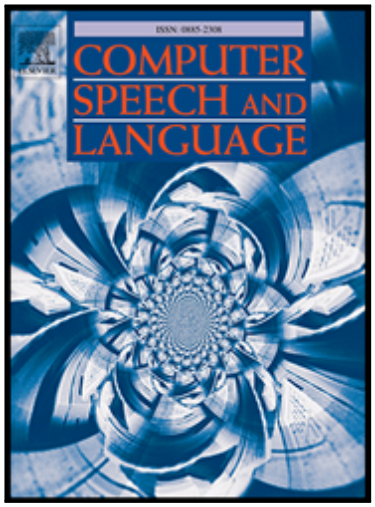
Journal Pre-proof



Sequence Labeling to Detect Stuttering Events in Read Speech

Sadeen Alharbi, Madina Hasan, Anthony J H Simons,
Shelagh Brumfitt, Phil Green

PII: S0885-2308(19)30296-7
DOI: https://doi.org/10.1016/j.csl.2019.101052
Reference: YCSLA 101052

To appear in: *Computer Speech & Language*

Received date: 7 February 2018
Revised date: 18 November 2019
Accepted date: 25 November 2019

Please cite this article as: Sadeen Alharbi, Madina Hasan, Anthony J H Simons, Shelagh Brumfitt, Phil Green, Sequence Labeling to Detect Stuttering Events in Read Speech, *Computer Speech & Language* (2019), doi: https://doi.org/10.1016/j.csl.2019.101052

Highlights

- The effect of data augmentation technique has improved the performance of all applied classifiers.
- The results for the stuttering detection task on human transcripts (F 1 scores) show that, without feature engineering, the BLSTM classifiers outperform the CRF classifiers by 33.6%.
- The results after added auxiliary features to support the CRF_{aux} classifier allows performance improvements by 45% and 18% relative to the CRF baseline ($\text{CRF}_{\text{ngram}}$) and BLSTM results, respectively on human transcripts.
- The results of $\text{CRF}_{\text{ngram}}$, CRF_{aux} and BLSTM classifiers on ASR transcripts, scored against human transcription degrade in these three classifiers by 7%, 12% and 11% respectively.

Sequence Labeling to Detect Stuttering Events in Read Speech

Sadeen Alharbi[1], Madina Hasan[1], Anthony J H Simons[1], Shelagh Brumfitt[2], Phil Green[1]

[1] *Computer Science Department, The University of Sheffield,*
[2] *Human Communication Sciences Department, The University of Sheffield, Sheffield, United Kingdom.*

Abstract

Stuttering is a speech disorder that, if treated during childhood, may be prevented from persisting into adolescence. A clinician must first determine the severity of stuttering, assessing a child during a conversational or reading task, recording each instance of disfluency, either in real time, or after transcribing the recorded session and analysing the transcript. The current study evaluates the ability of two machine learning approaches, namely conditional random fields (CRF) and bi-directional long-short-term memory (BLSTM), to detect stuttering events in transcriptions of stuttering speech. The two approaches are compared for their performance both on ideal hand-transcribed data and also on the output of automatic speech recognition (ASR). We also study the effect of data augmentation to improve performance. A corpus of 35 speakers' read speech (13K words) was supplemented with a corpus of 63 speakers' spontaneous speech (11K words) and an artificially-generated corpus (50K words). Experimental results show that, without feature engineering, BLSTM classifiers outperform CRF classifiers by 33.6%. However, adding features to support the CRF classifier yields performance improvements of 45% and 18% over the CRF baseline and BLSTM results, respectively. Moreover, adding more data to train the CRF and BLSTM classifiers consistently improves the results.

Keywords: Stuttering event detection, Speech disorder, CRF, BLSTM

1. Introduction

Stuttering, also known as stammering, is a speech communication disorder that can have severe social, educational and emotional maladjustment consequences, not only for the people who stutter but also for their families [1, 2]. It is presumed that early intervention is best to offset potential later impacts of having a stutter on one's psycho-social and communication developments [3]. During the assessment phase, clinicians carefully measure the stuttering events to determine if the stuttering is normal disfluency, borderline stuttering or beginning stuttering [4]. There are several approaches to determine stuttering severity. The fluency of very young children is commonly assessed through a conversational task, whereas for children older than seven years, a reading task may be used [5, 6]. The clinician asks the child to read from a passage, and then records each instance of disfluency while the child is reading. Clearly, this process is extremely dependent on the clinician's experience [7, 8, 9, 10].

In another approach, which constitutes a more accurate diagnostic method [11], the clinician transcribes a recorded session and classifies each spoken word according to several stuttering categories (including different kinds of repetition, prolongation, blocks and interjections) [5]. Having a literal transcription of a patient's speech can facilitate the detection of different types of stuttering event. In addition, archived transcriptions are useful for further investigative research into the condition. However, recording and then manually transcribing the stuttering speech is expensive, tedious and requires time and effort due to the need to chronicle each spoken word. Automating the transcription of the recorded speech using automatic speech recognition (ASR) could expedite the assessment of children's speech and make it easier to archive the data for further evaluation. This motivates the need for an ASR system that produces word level transcriptions, and a classifier that detects stuttering events in the acquired transcriptions. In our current study, we focus on a reading task because it is easier for the ASR system to recognise read speech, since the ASR has a prior knowledge of what the child intends to say, which limits the search space

for predicting the next word.

Automatically recognising children's speech is a well-known challenge, due to several factors, such as speech spontaneity, slow rates of speech, variability in vocal effort and the fact that children have smaller vocal tracts than adults [12, 13]. Attempting to distinguish and detect stuttering events in children's speech adds to the complexity of this task. In our proposed work, we are attempting to help therapists by providing them with an indication of the severity level of stuttering from an audio speech recording. The starting point for this is to adapt an ASR system to recognise stuttering events and provide a full-verbatim transcription. Then, the ASR output can automatically be processed to analyse the detected stuttering events. The novel contribution in this research has been to compare the performance of conditional random field (CRF) and bi-directional long-short term memory (BLSTM) classifiers in detecting stuttering events in both human and ASR transcripts of children's read speech. This comparison is conducted using lexical and contextual features.

Furthermore, this study investigates the effect of augmenting the available training data with artificially-generated training data to improve the performance of the classifiers. Finally, this work describes a method for studying the effect of ASR errors on the performance of the classifiers. The rest of this paper is organised as follows. Section 2 presents a review of the related literature. Section 3 describes the guidelines and methodology used to produce the stuttering data transcriptions and annotations. Section 4 describes the CRF and BLSTM classification approaches. Section 5 presents the feature engineering and extraction processes. Section 6 provides a description of the ASR system used in the study. Section 7 presents the experimental setup and results. Section 8 presents the conclusion and recommendations for future research.

2. Previous Work

Several disfluency correction systems have been introduced in the past [14, 15, 16, 17, 18]. These systems focus on the elimination of disfluent events in

order to generate more fluent outputs from speech recognition, with a view to providing better quality input to modules that perform downstream language processing. In contrast, our aim is to detect and classify stuttering events explicitly to facilitate the counting of each event.

As mentioned above, in one of the manual stuttering assessment approaches, the clinician transcribes a recorded session and classifies each spoken term into one of several normal, disfluent or stuttering categories [4, 5, 11]. If we start from the premise that the speech has already been transcribed by the therapist or the proposed ASR, the task is then to detect and classify stuttering events within the transcriptions. Mahesha and Vinod [19] used a lexical rule-based (RB) algorithm to detect stuttering events in orthographic transcripts from the University College London Archive of Stuttered Speech (UCLASS) [20]. In particular, they used prior domain knowledge to construct expert-based sets of rules to count the number of occurrences of each stuttering event. For event detection tasks, the traditional RB algorithm is a powerful tool for transferring the experiences of domain experts enabling them to make automated decisions. However, this approach depends on the expert's knowledge being complete, the rules fully covering every possible stuttering event, and articulation of the rules supporting diagnosis without rule-conflicts [21]. Previously [22], we proposed using a probabilistic approach that applies the machine learning classifiers HELM (Hidden Event Language Model) and CRF (Conditional Random Field) to the task of detecting stuttering in transcriptions of continuous children's speech. HELM and CRF are sequence labelling classifiers, since the probabilistic rules learned by these classifiers are entirely data-driven. Experimental results show that the CRF classifier outperforms the HELM classifier by 2.2% [22]. Our present study demonstrates an evaluation of two machine-learning approaches, CRF and BLSTM for detecting stuttering events both in human and ASR transcripts of children's read speech.

While detecting stuttering in human transcripts is useful, in many cases such transcripts may not exist. Our eventual goal is to detect stuttering in ASR transcripts. Some studies have investigated how ASR errors affect classification

results [16]. It is well known in the literature that, for different classification tasks, the performance of classifiers decreases in the presence of ASR errors. For example, errors in the ASR output caused a degradation in the performance of the Hidden Markov Model (HMM) and the Maximum Entropy Model (MaxEnt) by 50.6% and 52%, respectively in a disfluency detection task, using the broadcast news (BN) speech corpus [16].

3. Data Transcription and Annotation

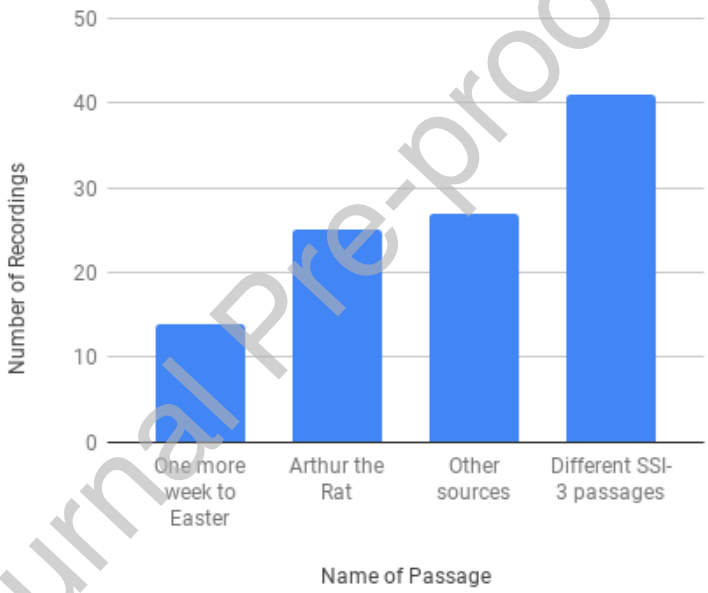


Figure 1: Histogram of different reading passages included in the UCLASS, Release Two dataset. See main text for an explanation of the passages.

The present study is based around a standard reading task that is used by therapists to diagnose stuttering in children. UCLASS, Release Two provides 42 texts read by children from the stuttering severity instrument (SSI-3) text readings, which are suitable for the age of the participants [6]. Another 25 recordings feature the reading of a passage from 'Arthur the Rat' [23], 14 recordings include a passage from 'One more week to Easter', a text developed by UCL,

and 27 come from other sources. Figure 1 illustrates the relative frequency of recordings of each passage in the corpus.

Table 1: Overview of datasets taken from each release of the UCLASS corpus. Age range, mean and standard deviation (sd) of the age for applied recordings in each category are given in NNyNNm format where y is year and m is the month.

Dataset	Age			Gender	
	Range	Mean	sd	Male	Female
Release 1 (monologue)	7y7m-17y9m	12y5m	2y0m	45	18
Release 2 (read speech)	8y4m-18y1m	12y6m	1y9m	48	0

For training purposes, we supplemented the read speech dataset from UCLASS, Release Two with a spontaneous stuttering speech dataset from UCLASS, Release One [20]. Table 1 provides a detailed breakdown of these subsets. We used 48 recordings of children's read speech (Read), taken from 48 males aged between 8 and 18 years, and 63 recordings of spontaneous speech (Spon) taken from 45 males and 18 females aged between 7 and 17 years, giving a total of 111 files from the UCLASS corpus.

Whereas 31/63 recordings from Release One already had orthographic transcriptions, there were no transcriptions for Release Two. We transcribed the rest of the Release One data following the same conventions and applied the same approach to the Release Two dataset. Transcriptions were orthographic, and included conventional forms to represent stuttering dysfluencies, for example: *This is a a a amazing.* Transcriptions were checked by a UK registered speech-language pathologist to ensure inter-annotator agreement.

The manual transcriptions of this data were later used to build a language model. Since this combined dataset was still relatively small, we also examined the effect of data augmentation, adding artificially generated transcription data (Art) designed to increase the likelihood of stuttering events in the language model.

The present study used the SRILM toolkit [24] to generate additional stut-

tering sentences from two inputs: a language model, trained on the UCLASS Release One training set, and a large word list. This word list was created by merging the UCLASS Release One word list with another publicly available word list lm-csr-64k-vb-3gram [25] which we augmented by rule with stuttering events, using a script that systematically generated stuttering events from existing words, adding all unique sound repetition, part-word repetition, prolongation and interjection forms to the word list.

The original transcription files for the recordings obtained from UCLASS (111 files from read and spontaneous tasks) and the artificial sentences generated by SRILM from the augmented data were then annotated with a label indicating the stuttering type for each word, using the labelling approach proposed by Yairi and Ambrose [4]. This constituted the training data for the different machine learning algorithms.

The stuttering events were identified and words annotated with special symbols, corresponding to the eight types of stuttering deemed significant by Yairi and Ambrose [4]:

1. sound repetitions, which include phone repetitions of less than one syllable (e.g. '*fa face*');
2. part-word repetitions, which refer to a repetition of less than one word and one or more complete syllables (e.g. '*any anymore*');
3. word repetitions in which an entire word is repeated (e.g. '*mommy mommy*');
4. prolongations, which involve an inappropriate duration of a phoneme sound (e.g. '*mmmay*');
5. phrase repetitions that repeat at least two complete words (e.g. '*it is it is*');
6. interjections (this term is used in the stuttering literature; in ASR these are often known as 'fillers'), which involve the inclusion of meaningless words (e.g. '*ah,um*');
7. revisions that attempt to fix grammar or pronunciation mistakes (e.g. '*I ate; I prepared dinner*'); and

8. blocking, which involves a stoppage of sound (any halting of speech, not just glottal stops) that can be momentary or longer and which occurs at an inappropriate place in an utterance, often including localised vocal tension.

Table 2: Types of Stuttering

Label	Stuttering Type
I	Interjection
S	Sound repetitions
PW	Part-word repetitions
W	Word repetitions
PH	Phrase repetitions
P	Prolongation
NS	Non-stutter

Six of the types of stuttering examined in this study (excluding blocking and revision, for reasons of tractability) are listed along with their corresponding abbreviations in Table 2. The stuttering annotation methodology was reviewed by a UK registered speech-language pathologist to ensure that correct judgments had been made about stuttering events. The transcribed text was also normalised, to ensure that special text entities such as dates, numbers, times and currency amounts were converted into words, as a prerequisite to downstream processing tasks.

In order to evaluate the ability of machine learning approaches to detect stuttering events from transcriptions obtained from a read task, we partitioned the transcribed 48 recordings of read data into training (80%) and evaluation (20%) sets, and we deliberately ensured that the training and evaluation sets had relatively equal distributions of stuttering events (Table 3). We trained initially only using the read data (*Read*), then on the read and spontaneous data (*Read+Spon*), then on the artificial data (*Read+Spon+Art*). Statistics for the training sets that included read/spontaneous tasks and a third training set

with artificial stuttering events are also shown in Table 3. Table 4 presents the distribution of each type of stuttering event in the evaluation set.

Table 3: Statistical Data for the Training Sets

Task	Training Data	Words	%I	%W	%PW	%S	%PH	%P	%NS
$Task_1$	Read	13134	0.22	1.9	0.33	1.9	1.5	1.3	93.6
$Task_2$	Read + Spon	24137	1.8	1.9	1.0	5.3	1.2	1.4	87.4
$Task_3$	Read + Spon + Art	74198	1.8	1.8	1.4	4.0	1.0	1.0	89
Avg		37156.3	1.3	1.9	1.00	3.7	1.2	1.2	90

Table 4: Statistical data for the test set from the human transcripts

Set	Words	%I	%W	%PW	%S	%PH	%P	%NS
Test	3189	0.3	1.2	0.7	1.00	1.6	0.44	94.8

4. Detecting Stuttering Events

4.1. Task Definition

Given a sentence (sequence of proper/stuttered word entities), the task is to assign a stuttering label (I, W, PW, S, PH, P or NS) to each entity in the sentence. Some entities have a unique label, such as the interjection words. Others could have different labels that vary with the location in the sentence.

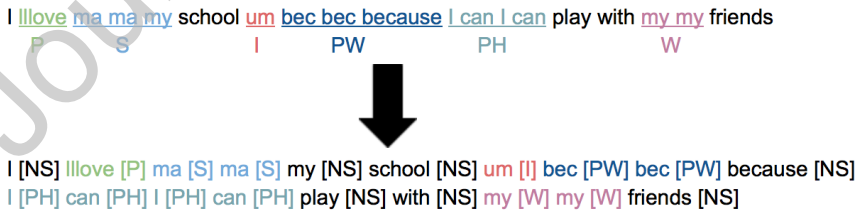


Figure 2: An example of the annotation process. Above is an example of an input sentence, indicating regions of stuttering for illustrative purposes only. Below is an example of how each word of the sentence was actually annotated by the described approach.

Figure 2 illustrates some examples of these situations. In this way, determining the correct stuttering label for each entity depends on the entity's context within the sentence, including the labels of the neighbouring entities. Therefore, the task of detecting stuttering events in a transcription can be defined as a sequence labelling task and approaches such as Conditional Random Field (CRF) and Bi-directional Long-Short Term Memory (BLSTM), can be used. The task can be formulated as follows: given a sequence of observations/feature vectors, find an appropriate label sequence for the observations. The following section describes the CRF and BLSTM approaches used in the present study.

4.2. Conditional Random Fields

Conditional Random Fields (CRFs) are a class of linear statistical models which are known to exhibit high performance in sequence labelling tasks [26, 27, 28, 29]. This is because a CRF classifier takes into account the probability of co-occurrence between neighboring labels and simultaneously estimates the best sequence of predicted labels for a given input sentence. Following an approach similar to that taken for the named entity recognition (NER) task [30], we created a CRF model for the stuttering event detection task by designing different features to detect and classify sound, word, and phrase repetition (explained in Section 5) which observes the entire represented region.

This model also aims to estimate and directly optimise the posterior probability of the label sequence, given a sequence of features (hence the frequently used term *direct model*). In particular, given a set of observations (a sequence of words that may include some stuttered words), a CRF model predicts a sequence of labels y for these observations. Let X,Y be the observation and label sequences, respectively, and $f(x, y)$ be the set of feature functions. The CRF model can be represented by the following equations:

$$p(y|x, \lambda) = \frac{exp(\lambda^T f(x, y))}{Z(\lambda, x)},$$

where λ is the model's parameters and Z is the normalisation term. One weight (w) is determined for each feature. These weights are learned during training

such that:

$$\lambda = \arg\max_{\lambda} p(Y|X, \lambda),$$

and the label sequence can then be predicted from the following equation:

$$y^* = \arg\max_{y} p(y|x, \lambda) argmax_y p(y|x, w).$$

4.3. Bidirectional Long Short-Term Memory

Recently, a class of neural networks trained on word representations with a distributed input (known as *word embeddings*) have been widely used in problems related to natural language processing (NLP) with great success [31, 32, 33]. Long Short-Term Memory (LSTM) units were proposed by [34]. These are a variation of the Recurrent Neural Networks (RNNs) that are able to capture long-term dependencies with the guidance of the particular structure [35]. The input for the LSTM is a sequence of word embeddings $w_1, \ldots, w_M$, where $w_i \in V$, the vocabulary list, and the output is a sequence of events T. Where $\hat{y}_i$ denotes the predicated event (e.g I, S, PW, W, PH, P or NS) for word w_i, $\hat{y}_i$ can be predicted using the softmax activation function.

In the stuttering labelling task, it is important to capture the dependency both on past features, and also on future features, which together characterise different kinds of stuttering event, such as sound, word, part-word, and phrase repetition. We therefore employ the bidirectional LSTM structure (BLSTM) used by [36], which allow us to apply the forward and backward steps, so making best use of both past and future features.

LSTM units are a variation of the RNN that overcome the problem of vanishing gradient [35]. This property allows LSTMs to capture long-term dependencies without arithmetic problems. In particular, LSTM incorporates gated memory cells by which signals can be read, written, deleted or stored. These operations are controlled using a sigmoid function that performs element-wise operations. The decisions in this process are based on weights that are learned during the recurrent network training. Figure 3 illustrates the data flows in a

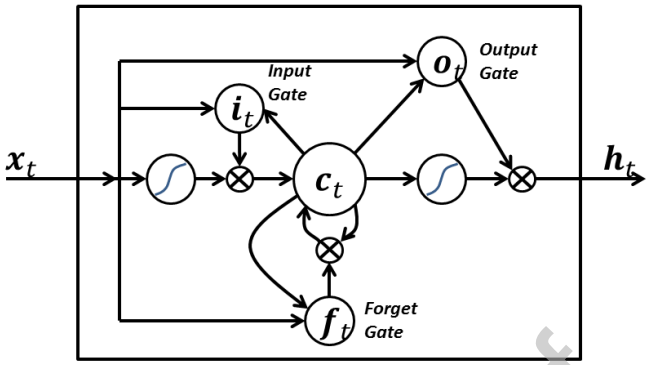


Figure 3: *Long short-term memory cell. This figure presents the data flows in a memory cell where f_t, c_t, i_t, o_t, h_t are the forget, cell, input, output gates and hidden state, respectively. X_t is the input vector at the time t* [36].

memory cell. The different LSTM gates can be modelled using the following formulae to update an LSTM unit at time t [37]:

$$
\begin{aligned}
i_t = &\quad \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i),\\
f_t = &\quad \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f),\\
c_t = &\quad (f_t c_t - 1) \odot c_{t-1} + i_t \odot tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c),\\
o_t = &\quad \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o),\\
h_t = &\quad o_t \odot tanh(c_t),
\end{aligned}
$$

where f_t, c_t, i_t, o_t, h_t are the forget, cell, input, output gates and hidden state, respectively. X_t is the input vector at the time t. W and b are the weights and the biases vectors of the network, respectively. σ is the element-wise sigmoid function and $\odot$ is the element-wise product. To form the final BLSTM representation, the left-to-right $\overrightarrow{h_t}$ and the right-to-left $\overleftarrow{h_t}$ input representations are concatenated $ht = [\overrightarrow{h_t}; \overleftarrow{h_t}]$.

5. Features of the Classifiers used to Detect Stuttering Events

Assigning a label to a word entity is based on a set of observations, associated with this label. These observations are introduced to a classifier as a set of

feature vectors. The role of the classifier is to map the set of feature vectors to a specific label, and this is done by implementing a set of steps that varies with different classifiers.

This section describes the features used by the proposed classifiers to detect the stuttering events in transcriptions. This includes uni-gram, bi-gram, tri-gram and 2-post-words for each word, as well as character- and utterance-based features for the CRF classifier and pre-trained word embeddings for the BLSTM classifier.

5.1. Word/Utterance-Based Features

This work introduces long-range statistics by measuring the backward distance at two levels. The first level uses a backward distance metric to count how far the current (pseudo-) word is from the first of a sequence of identical (pseudo-) words repeated in the word sequence. For example,

sequence	*sa*	*sa*	*sa*	*sound*
distance	0	1	2	0

This feature aids the classifier to observe the repeated patterns in the text. At the second level, we also measure and compare the backward distance of each neighbouring bigram and trigram word group. For the example: (*it is it is*) the assigned counter for the second *it* and *is* will be 1, which is an indication of phrase repetition of a bigram phrase.

5.2. Character-Based Features

This kind of feature was also extracted at two levels. The first level uses a backward distance to measure how far the current character is from the first of a sequence of identical characters repeated within the same word. For example, (*mmmay*) assigns counters 1 and 2 to the repeated occurrences of the prolonged *m*. This feature helps the classifier to observe the kind of character repetition which indicates a prolongation event. At the second level, we measure and compare the backward distance between identical groups of two and three characters, over successive words or part-words. For example, in (*par particular*),

the assigned counter for the characters of the second pseudo word *par* within word *particular* will be 1,1,1.

5.3. Word Embedding

A meaningful word representation can be learned using neural networks from random word embedding. Word embedding is employed by converting each word into a numerical vector in a vector space by assigning all semantically-similar words to similar vectors. A well-known algorithm, GloVe [38], was used to perform the word embedding technique. This algorithm builds word embeddings by searching through the training data to find co-occurrences of words with the assumption that the meaning of a word usually depends upon its context. In the present study, we used a pre-trained GloVe model to generate word embeddings for each utterance. This model was trained on the Common Crawl (CC) corpus (1.9 M vocab) [38].

6. Automatic Speech Recognition System

Applications of ASR in the domain of speech pathology, as a form of assistive technology for therapsists, have appeared in recent years. Adapting an ASR system for developing clinical technologies to provide an automatic assessment could assist speech-language pathologists (SLPs) in providing a better service [39, 40]. This could be in the form of an initial triage, or as part of a fully automated diagnosis.

In the current work, We used the Kaldi ASR toolkit [41] to build and train an ASR system. To increase the likelihood that the ASR would detect stuttering events, we augmented the ASR's language model with artificially generated stuttering data. The ASR was then able to recognise different stuttering events in the continuous speech of children and also produced a useful word-level transcription of what was said (henceforth ASR transcripts, to distinguish these from clinician's transcripts).

The ASR had one limitation. Its Hidden Markov Model (HMM) yielded a time-based segmentation as a consequence of matching MFCC features in the

frequency domain; time was a derived property. We did not expect the HMM to detect time-sensitive stuttering events, such as prolongation, with any accuracy. We investigated using an HMM with explicit duration models, but there was insufficient training data (a common problem in speech pathology domains). In later work, we developed a separate autocorrelation method for time-sensitive events, that was trained in parallel to make best use of the data.

6.1. Speech Corpus and Transcription

The ASR system had to be capable of identifying stuttering in the speech of children. The chief challenge was the small amount of children's stuttering data available for training. We used data sets from the University College London Archive of Stuttered Speech (UCLASS) [20]. Data from UCLASS Release 2 (read speech) was added to a much larger corpus of children's read speech, the PF-STAR Children's Speech Corpus [42], to satisfy training requirements. This data contained mostly fluent speech and non-stuttering kinds of disfluency found in children's speech.

The PF-STAR sentences were already transcribed; and we undertook the transcription of the UCLASS data as described in Section 3. Where we diverged slightly from the orthographic style used in UCLASS, Release One, was in the notation devised for sound repetition. The original convention had used consonant-repetition, such as *w w what*; but experimentally we found that this could not be interpreted consistently by the pronunciation dictionary used by the ASR. Instead, we adopted a CV convention, such as *wa wa what*, where the orthographic vowel was mapped to the phonetic vowel schwa in the pronunciation dictionary. Empirically this enabled good recall of sound repetition.

Only 48 speech samples of readings (of approximately two hours) were used in the current research. A cross-validation technique was employed to partition the available data, such that a different subset was used as the test set and the rest for training on each rotation, as is usual with a small dataset.

6.2. Language Model Augmentation

The performance of an ASR system is highly dependent on the amount of text included in the training corpora. In general, richer text results in a better trained model. However, the style of text used for training the model also needs to match closely the language style expected in the ASR application. In our case, we had considerably fewer examples of stuttering in our training data compared to fluent speech. An approach called language model augmentation may be used to address this, which generates additional artificial data for the kinds of events not commonly reflected in the available training data. The artificial data have to be generated from another source of knowledge that is capable of providing relative frequencies of the artificial events that more closely match their probabilities in the target language we wish to recognise.

The augmentation approach was based around the existing segmentation of the UCLASS transcriptions into short utterances, each typically a few words in length (used in alignment). We augmented each utterance multiple times, by inserting an artificial sound repetition at the start of each word in turn, adding this set of utterances to the language model. For example, where the original data has '*come down*', we added '*ca come down*' and '*come da down*' (using the orthographic conventions described above). We speculated that this would not only increase the frequency of sound repetitions (the least well represented category), but also the frequency of other stuttering events already present in the same utterances, by virtue of data duplication.

The augmentation rule mimicked repeating the syllable-onset of the following word. Orthographically, we used the initial consonant of the following word, followed by a schwa, based on Howell and Vause's observation [43] that schwa is the most commonly inserted vowel. They give an articulatory explanation for this [44]. We also found empirically that using schwa everywhere (even when the onset-vowel was sometimes coloured) gave better recall performance than using different weak vowels. Adding the extra augmented utterances as above also tended to increase the frequency of different repetition and interjection events, for which there were already sufficient incidences in the training data.

The results show that the performance of the ASR in discriminating stuttering from non-stuttering events improved in recall from 38% to 73%, improved in precision from 57% to 84% and reduced the average WER from 19.8% to 15.9%. We previously described this ASR system in [45]; but the focus of the current work is on processing the transcriptions generated by it.

7. Experiments

The first experimental sets were designed to compare the performance of the CRF and BLSTM classifiers in relation to human transcripts. We selected varying amounts of training data from the different speaking tasks and studied how using increasing amounts of data affected the performance of the classifiers. These investigations were conducted with and without the proposed characters/word and utterance features for the CRF classifier, presented in Section 5. In addition, we studied how the classifiers are affected by speech recognition errors, such as deletions, substitutions and insertions.

For the evaluation, we use orthographic transcriptions of the test sets as references. These references were labeled manually by clinicians, following Yairi and Ambrose's annotation approach [4].

7.1. CRF classifier: Effect of Adding More Data

The baseline CRF classifier was trained using word $n-$grams, where $n = 1, 2, 3$, and two post-word features were extracted from the read speech data. There were three sets of experiments. In Task1, only the read speech data was used. In Task 2, this was supplemented by the spontaneous speech data; and in Task 3, was further supplemented by artificial data. The details of the artificial data are given in Section 3 and the distribution of stuttering events in all used data are presented in Table 3.

7.1.1. CRF using only ngram features

Table 5: CRF_{ngram} results, when trained on different tasks, using ngram features only. Task_1 used only the (*Read*) data. Task_2 used the (*Read+Spon*) data. Task_3 used the (*Read+Spon+Art*) data. The column "St" describes the stuttering type.

St	Task_1			Task_2			Task_3		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
I	1.00	0.44	0.62	1.00	0.56	0.71	1.00	0.67	0.80
W	1.00	0.50	0.62	1.00	0.50	0.62	1.00	0.50	0.62
P	1.00	0.14	0.25	1.00	0.14	0.25	1.00	0.14	0.25
PH	0.52	0.27	0.36	0.65	0.25	0.37	0.56	0.23	0.33
PW	1.00	0.10	0.17	1.00	0.05	0.09	1.00	0.00	0.00
S	1.00	0.59	0.74	1.00	0.83	0.91	1.00	0.93	0.96
Average	0.92	0.34	0.49	0.94	0.39	0.55	0.93	0.41	0.57

Table 5 shows the results of three CRF classifiers on the detection of each type of stuttering event. These classifiers differ in the type/task and amount of data used for training, as shown in Table 3. One can clearly observe that, with the addition of stuttered spontaneous data, the precision results have either improved (PH by 20%, relatively) or not changed for all stuttering types. The recall results have either improved (I by 21.4%; S by 28.9%) or deteriorated (PH by 7.4%; PW by 50%), resulting in the average F_1 measure for those labels. These results can easily be interpreted when linked to the change in the distribution of each class, after adding the spontaneous speech data. In particular, improvement in the detection of I and S is linked to the increase in their distribution (I by 87.8%; S by 64.2%), as shown in Table 3. However, the deterioration in detecting PH is due to this being a lower proportion of the larger training set in which its distribution was reduced by 20%.

Similarly, the PW detection was not improved. This is mainly due to the fact that words in this category have a wide range of variation because they are literally sub-units of the words. For instance, in cases, such as '*pla* [PW], *play*

[PW]', which only occurs in the evaluation set, the CRF will not be able to detect it because it was not seen in the training set (viz. out of the domain vocabulary, OOV). The results in Table 5 also suggest that adding more data, without additional features, will not improve the detection of *PW* events unless the additional data contains the *PW* words that are in the evaluation set. Moreover, adding more data that affect the distribution of other classes may negatively affect the detection of these types of word-dependent classes. On average, adding spontaneous data increased the overall averages of all the measures (recall by 12.8%, precision by 2%, F_1 by 10.9%).

The last set of results shows the outcome of adding artificial data (Table 5). The only class that benefits from this addition is the sound repetition class. This result is expected from the distribution reported in Table 3, which shows improvement only in the *S* and *PW* classes. The *PW* deterioration is due to the same reason discussed earlier, and the only solution for this type of dependency is to use additional features to help the classifier detect unseen part-words. These results motivated the introduction of context-based features to reduce the dependency on the word fragments represented in n-grams and post-word features. This is addressed in the next section.

7.1.2. CRF using extra character- and utterance-based features

Table 6: CRF_{aux} results, when trained on different tasks, using auxiliary features. Task_1 used only the (*Read*) data. Task_2 used the (*Read+Spon*) data. Task_3 used the (*Read+Spon+Art*) data. The column "St" describes the stuttering type.

St	Task_1			Task_2			Task_3		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
I	1.00	0.78	0.88	1.00	0.78	0.88	1.00	0.78	0.88
W	0.95	1.00	0.97	0.95	0.97	0.96	1.00	0.97	0.99
P	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
PH	0.72	0.72	0.72	0.72	0.72	0.72	0.88	0.70	0.77
PW	0.63	0.57	0.60	0.53	0.71	0.63	0.82	0.94	0.87
S	0.96	0.79	0.87	1.00	0.93	0.96	1.00	1.00	1.00
Average	0.88	0.81	0.84	0.86	0.85	0.85	0.95	0.88	0.92

As shown in Table 6, the performance of the proposed CRF classifier with the word feature and the character/utterance-based features (described in Section 5) has improved over all previous stuttering types, leading to better results in the classification task. For critical classes that mainly depend on word representations, such as *PW* and *P*, adding character-based features helped detect all words with repeated characters *P*, such as '*mmmay*'. Moreover, this feature enhanced the PW classification so that it achieved a 60% F_1 score on the read task, a 63% F_1 score on the mixed read/spontaneous task and an 87% F_1 score after adding more artificial stuttering data. Adding an utterance-based feature improved the ability of the CRF classifier to detect phrase repetitions by 36% in comparison to the baseline experiment. Adding a word-distance feature enhanced word repetition *W* detection by 37% in comparison to the baseline.

7.2. Bidirectional Long Short-Term Memory

Table 7: BLSTM results, when trained on different tasks, using embedded features. Task_1 used only the (*Read*) data. Task_2 used the (*Read+Spon*) data. Task_3 used the (*Read+Spon+Art*) data. The column "St" describes the stuttering type.

St	Task_1			Task_2			Task_3		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
I	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
W	1.00	0.05	0.10	0.75	0.08	0.14	0.77	0.29	0.43
P	0.89	0.57	0.70	1.00	0.71	0.83	1.00	0.71	0.83
PH	0.88	0.57	0.69	0.73	0.71	0.72	0.74	0.67	0.70
PW	0.00	0.00	0.00	0.71	0.24	0.36	0.50	0.50	0.50
S	1.00	0.79	0.88	0.93	0.86	0.89	1.00	1.00	1.00
Average	0.80	0.50	0.61	0.85	0.60	0.70	0.84	0.70	0.76

We trained the BLSTM classifier with word embeddings extracted from a pre-trained GloVe model (Common Crawl, 1.9 M vocab) [38]. This experiment examined how adding data would affect the performance of the BLSTM classifier. The hyperparameters of the BLSTM were tuned on a development set; the number of hidden nodes was 50, with a word embedding dimension of 300, a learning rate of 0.001 and a drop-out rate of 0.5. All weights in the network were initialized randomly from the uniform distribution within range [-1, 1]. The number of training "epochs" (i.e. iterations) was set to 30. The Tensorflow neural network toolkit [46] was used for BLSTM implementation. The results presented in Table 7 show how adding spontaneous and artificial data to the read data improves the results. When only using a reading task to train the BLSTM, its performance in detecting stuttering events was very low, especially for W and PW. This is expected, due to the small training data set that was used. After adding the spontaneous and artificial data, this significantly improved the performance of the BLSTM classifier. It outperformed the CRF baseline classifier, benefiting from its embedded word representation to group

multiple occurrences of similar tokens (whereas CRF used word n-grams).

In terms of computation-time taken to train BLSTM networks for the three tasks, the largest training set applied was in Task3 (a training set of 74,198 words). For this, the average time taken to train each epoch was 39.6 seconds. On the other hand, the time needed to train the CRF model was approximately 4 seconds. Adding auxiliary features to train the CRF model increased the computation-time to 11 seconds.

Table 8: Summary table of the best results for classifying human transcriptions, using classifiers trained on Task_3. The column "St" describes the stuttering type.

St	CRF_{ngram}	CRF_{aux}	BLSTM
	F_1	F_1	F_1
I	0.80	0.88	1.00
W	0.62	0.99	0.43
P	0.25	1.00	0.83
PH	0.33	0.77	0.70
PW	0.00	0.87	0.50
S	0.96	1.00	1.00

In general, with the increase of training data in the three applied tasks, the average computation-time increased, but the F-score for stuttering classification was improved. The best results are presented in Table 8 and this was obtained on Task_3. The results obtained from CRF_{aux} were higher than those obtained from BLSTM. However, BLSTM results without using feature engineering were still considered high and comparable with CRF_{aux}.

7.3. Evaluation on ASR transcripts

The set of experiments presented in the previous sections evaluated the performance of the classifiers in relation to an ideal labelling of a clinician's orthographic transcriptions with stuttering events. However, as mentioned above, our goal is to detect stuttering in audio recordings directly, bypassing the need for

expert transcription altogether. Consequently, the performance of the classifiers was also evaluated on ASR transcriptions.

The output from the ASR was error-prone, introducing insertions, deletions and substitutions with respect to the reference transcriptions. Moreover, ASR systems tend to delete non-word entities, such as '*um*' or '*ah*' fillers (interjection), which we would prefer to preserve as stuttering events. Similarly, an artifact of having trained the ASR with an augmented language model meant that the ASR was more likely to insert false positives for some stuttering events, which could confuse the classifiers.

7.3.1. Error propagation

These errors propagate to the stuttering detection stage and affect the performance of the classifiers in two ways. Firstly, there is a mismatch between the data used to train the classifiers (clinician's transcripts) and the evaluation set (ASR transcripts). Secondly, the ASR creates different types of error (i.e. deletion, insertion and substitution errors). The classifiers will nonetheless label these errors as stuttering events and, even if these labels are correct with respect to the ASR transcript, they might be incorrect with respect to the clinician's reference transcript. For example, the classifier will label any repeated word that is inserted by ASR as *W* because it is a word repetition. However, such words were inserted by the ASR and do not exist in the reference, which leads to the generation of an incorrect stuttering event label in comparison with the clinician's reference text.

In our experiments, the ASR transcripts of the evaluation set have a WER of 12.4% with 1.6% insertion, 3.4% deletion and 7.3% substitution errors. These types of error affect the stuttering pattern or word fragments learned by the classifiers. The following two examples are taken from the evaluation set and illustrate this argument (Figure 4):

id:(m_1216_11y1m_1-70.153247-72.281907)						id: (m_1214_11y8m_1-53.864596-56.7129)						
Scores: (#C #S #D #I) 4 0 0 1						Scores: (#C #S #D #I) 6 1 1 0						
REF	***	the	storm	fighting	her	i	must	ha	have	got	a	fish
HYP	the	the	storm	fighting	her	i	must	**	have	caught	a	fish
Eval	I	C	C	C	C	C	C	D	C	S	C	C
REF$_{Label}$	***	NS	NS	NS	NS	NS	NS	S	S	NS	NS	NS
HYP$_{Label}$	W	W	NS	NS	NS	NS	NS	**	NS	NS	NS	NS

a)Insertion error example b)Deletion error example

Figure 4: (a) illustrates the effects of insertion errors on the classifier performance using the NIST scoring tool. In this case, due to the inserted word *the* followed by a correct word *the*, any perfect classifier would label both the actual and the repeated *the* words with the label W, which is incorrect with respect to the reference transcript, which has one *the*. (b) illustrates the effects of deletion errors on the classifier performance using the NIST scoring tool. In this case, due to deleting the sound repetition *ha*, the classifiers never saw the deleted sound, and mislabelled the following word as NS, which is incorrect with respect to the reference transcript.

These observations agree with the conclusions reported in the literature that a classifier's performance degrades when applied to error-prone transcripts for different ASR post-processing tasks, such as the deterioration reported in a disfluency detection task [16].

7.3.2. Evaluation

The previous experimental results show that the best models were those trained on Task$_3$ with additional features on top of the n-gram features. Hence, only those models were applied to detect the stuttering events in the ASR transcripts. The performance of the classifiers is evaluated against the actual labelled human transcript reference. Moreover, our ASR engine failed to recognize all interjection and prolongation words in the test set (all the 9 interjection and 14 prolongation words were deleted by the ASR), hence I and P were excluded from the evaluation.

To study how classifiers are affected by speech recognition errors and how these errors propagate into the classifiers decisions, the results of the classifiers on ASR transcripts needs to be compared against the human reference. This

comparison is made slightly harder, as a result of misalignment of the two transcripts, resulting from the ASR's insertion and deletion errors. To address this, we follow the alignment procedure described in [16], in which hypothesized labels are mapped to reference labels using timing information provided by the NIST scoring tool [47].

Table 9: Results of classifiers trained on Task_3, evaluated on ASR transcripts.

St-type	CRF_{ngram}			CRF_{aux}			BLSTM		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
W	0.79	0.47	0.59	0.94	0.91	0.91	0.88	0.44	0.58
PH	0.35	0.23	0.28	0.65	0.73	0.69	0.68	0.57	0.62
PW	1.00	0.12	0.21	1.00	0.55	0.71	0.89	0.40	0.55
S	0.92	0.80	0.86	0.92	0.80	0.86	0.92	0.80	0.86
Average	0.77	0.41	0.53	0.88	0.75	0.81	0.84	0.55	0.66

Table 9 presents the results of CRF_{ngram}, CRF_{aux} and BLSTM classifiers on ASR transcripts, scored against human reference. The relative F1 degradation in the three classifiers (trained on Task_3) in comparison with the performance on true human transcripts, are 7%, 11% and 6% respectively. Despite the insertion, deletion and substitution ASR errors, this degradation in the classifiers performance is considered acceptable, compared to the degradation reported in the literature for a similar task [16].

8. Conclusion

In the current study, the CRF and BLSTM sequence-labelling approaches were used to detect and label stuttering events, both within a clinician's manual transcripts and within transcripts generated by automatic speech recognition. Variations in the performances of the CRF and BLSTM classifiers were studied by varying the task and the amount of training data.

When evaluated on the clinician's transcripts, the experimental F_1 results

show that, without feature engineering, the BLSTM classifiers outperform the CRF classifiers by 33.6%. However, adding auxiliary features to support the CRF_{aux} classifier allows performance improvements of 45% relative to the CRF baseline (CRF_{ngram}) and of 18% relative to the BLSTM results.

When evaluated on the ASR transcripts, the performance of all classifiers degrades after propagating ASR errors. The interjection has been excluded from this set of experiments due to its limited occurrences in the training data. These findings agree with other findings reported in the literature for a similar task [16]. Furthermore, we ascribe this degradation in performance firstly to the ASR errors and secondly to the mismatch between the data used to train the classifiers and the test data.

However, the downstream consequences of this degradation may be less significant, where the eventual goal is to classify stuttering severity. ASR insertions and deletions lead both to the over- and under-estimation of stuttering events compared to the ideal labelling of the clinician's transcription. In some cases, one stuttering event is substituted for another. Even if ASR errors lead to detecting fewer stuttering events than the clinician's reference, it should be possible to tune the thresholds used by a diagnosis tool, so long as the behaviour of the ASR is consistent.

The ultimate goal of this work is to investigate how speech technology and machine learning approaches could assist in the diagnosis of stuttering severity. Where no transcription exists of the recorded audio of a stuttering child, our approach may be used with ASR to transcribe, detect and classify stuttering events. Where an existing therapist's transcription exists, our approach may still be used to detect stuttering events automatically. Both kinds of support may be found helpful and convenient in providing an initial triage of stuttering severity. Eventually, the purpose of providing automated detection of stuttering severity is to offer remote diagnosis where the relevant clinical expertise is absent.

Our future work will focus on the final diagnosis stage, in which tallies of stuttering events are used to determine the severity of stuttering. We will explore the effect of classifier and ASR errors on the tuning of thresholds used

to classify patients as non-stuttering, borderline stuttering, or beginning stuttering. Other avenues to explore will include improvements to the ASR stage needed to reduce the WER; and the investigation of an alternative method to detect prolongation events, which were not tractable using the current approach.

References

[1] R. Hayhow, A. M. Cray, P. Enderby, Stammering and therapy views of people who stammer, Journal of Fluency Disorders 27 (1) (2002) 1–17.

[2] A. Craig, P. Calver, Following up on treated stutterers studies of perceptions of fluency and job status, Journal of Speech, Language, and Hearing Research 34 (2) (1991) 279–284.

[3] M. Jones, M. Onslow, A. Packman, S. Williams, T. Ormond, I. Schwarz, V. Gebski, Randomised controlled trial of the Lidcombe programme of early stuttering intervention, BMJ 331 (7518) (2005) 659. arXiv: http://www.bmj.com/content/331/7518/659.full.pdf, doi:10.1136/bmj.38520.451840.E0.
URL http://www.bmj.com/content/331/7518/659

[4] E. Yairi, D. M. Carrico, Early childhood stuttering, American Journal of Speech-Language Pathology 1 (3) (1992) 54–62.

[5] H. H. Gregory, J. H. Campbell, C. B. Gregory, D. G. Hill, Stuttering therapy: Rationale and procedures, Allyn & Bacon, 2003.

[6] G. D. Riley, A stuttering severity instrument for children and adults, Journal of Speech and Hearing Disorders 37 (3) (1972) 314–322.

[7] G. W. Blood, C. Mamett, R. Gordon, I. M. Blood, Written language disorders: Speech-language pathologists' training, knowledge, and confidence, Language, Speech, and Hearing Services in Schools 41 (4) (2010) 416–428.

[8] S. B. Brundage, A. K. Bothe, A. N. Lengeling, J. J. Evans, Comparing judgments of stuttering made by students, clinicians, and highly experienced judges, Journal of Fluency Disorders 31 (4) (2006) 271–283.

[9] N. J. Lass, D. M. Ruscello, M. D. Pannbacker, J. F. Schmitt, D. S. Everly-Myers, Speech-language pathologists' perceptions of child and adult female and male stutterers, Journal of Fluency Disorders 14 (2) (1989) 127–134.

[10] D. J. Brisk, E. C. Healey, K. A. Hux, Clinicians training and confidence associated with treating school-age children who stutter: A national survey, Language, Speech, and Hearing Services in Schools 28 (2) (1997) 164–176.

[11] B. Guitar, Stuttering: An integrated approach to its nature and treatment, Lippincott Williams & Wilkins, 2013.

[12] H. Liao, G. Pundak, O. Siohan, M. K. Carroll, N. Coccaro, Q.-M. Jiang, T. N. Sainath, A. Senior, F. Beaufays, M. Bacchiani, Large vocabulary automatic speech recognition for children, in: Sixteenth Annual Conference of the International Speech Communication Association, 2015, pp. 1611–1615.

[13] M. Russell, S. DArcy, Challenges for computer recognition of childrens speech, in: Workshop on Speech and Language Technology in Education, 2007, pp. 108–111.

[14] M. Honal, T. Schultz, Correction of disfluencies in spontaneous speech using a noisy-channel approach, in: Eighth European Conference on Speech Communication and Technology, 2003.
URL http://www.isca-speech.org/archive/eurospeech_2003/e03_2781.html

[15] M. Snover, B. Dorr, R. Schwartz, A lexically-driven algorithm for disfluency detection, in: Proceedings of HLT-NAACL 2004: Short Papers, Association for Computational Linguistics, 2004, pp. 157–160.

[16] Y. Liu, E. Shriberg, A. Stolcke, M. P. Harper, Comparing hmm, maximum entropy, and conditional random fields for disfluency detection., in: Interspeech, 2005, pp. 3313–3316.

[17] M. Honal, T. Schultz, Automatic disfluency removal on recognized spontaneous speech-rapid adaptation to speaker-dependent disfluencies, in: Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on, Vol. 1, IEEE, 2005, pp. I–969.

[18] S. Maskey, B. Zhou, Y. Gao, A phrase-level machine translation approach for disfluency detection using weighted finite state transducers, in: Ninth International Conference on Spoken Language Processing, 2006.
URL http://www.isca-speech.org/archive/interspeech_2006/i06_1886.html

[19] P. Mahesha, D. Vinod, Using orthographic transcripts for stuttering dysfluency recognition and severity estimation, in: Intelligent Computing, Communication and Devices, Springer, 2015, pp. 613–621.

[20] P. Howell, S. Davis, J. Bartrip, The university college london archive of stuttered speech (uclass), Journal of Speech, Language, and Hearing Research 52 (2) (2009) 556–569.

[21] H. Liu, A. Gegov, M. Cocea, Complexity control in rule based models for classification in machine learning context, in: UK Workshop on Computational Intelligence, Vol. 513, Springer, 2016, pp. 125–143.

[22] S. Alharbi, M. Hasan, A. J. Simons, S. Brumfitt, P. Green, Detecting stuttering events in transcripts of childrens speech, in: International Conference on Statistical Language and Speech Processing, Springer, 2017, pp. 217–228.

[23] D. Abercrombie, English Phonetic Texts.(1. Publ.), Faber & Faber, 1964.

[24] A. Stolcke, SRILM-an extensible language modeling toolkit, in: Seventh International Conference on Spoken Language Processing, 2002, pp. 901–904.

[25] K. Vertanen, CSR LM-1 language model training recipe (2006).
URL http://www.keithv.com/software/csr/

[26] H. Tseng, P. Chang, G. Andrew, D. Jurafsky, C. Manning, A conditional random field word segmenter for SIGHAN bakeoff 2005, in: Proceedings of the fourth SIGHAN workshop on Chinese Language Processing, Vol. 171, Citeseer, 2005.
URL https://www.aclweb.org/anthology/I05-3027

[27] L. Ratinov, D. Roth, Design challenges and misconceptions in named entity recognition, in: Proceedings of the Thirteenth Conference on Computational Natural Language Learning, Association for Computational Linguistics, 2009, pp. 147–155.

[28] A. Passos, V. Kumar, A. McCallum, Lexicon infused phrase embeddings for named entity resolution, arXiv preprint arXiv:1404.5367.

[29] G. Luo, X. Huang, C.-Y. Lin, Z. Nie, Joint entity recognition and disambiguation, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 879–888.

[30] A. McCallum, W. Li, Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons, in: Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4, Association for Computational Linguistics, 2003, pp. 188–191.

[31] Z. Huang, W. Xu, K. Yu, Bidirectional LSTM-CRF models for sequence tagging, arXiv preprint arXiv:1508.01991.

[32] J. P. Chiu, E. Nichols, Named entity recognition with bidirectional lstm-cnns, Transactions of the Association for Computational Linguistics 4 (2016) 357–370.

[33] Z. Hu, X. Ma, Z. Liu, E. Hovy, E. Xing, Harnessing deep neural networks with logic rules, arXiv preprint arXiv:1603.06318.

[34] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[35] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE transactions on neural networks 5 (2) (1994) 157–166.

[36] A. Graves, A.-r. Mohamed, G. Hinton, Speech recognition with deep recurrent neural networks, in: Acoustics, speech and signal processing (icassp), 2013 ieee international conference on, IEEE, 2013, pp. 6645–6649.

[37] F. A. Gers, N. N. Schraudolph, J. Schmidhuber, Learning precise timing with lstm recurrent networks, Journal of machine learning research 3 (Aug) (2002) 115–143.

[38] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.

[39] L. Ward, A. Stefani, D. Smith, A. Duenser, J. Freyne, B. Dodd, A. Morgan, Automated screening of speech development issues in children by identifying phonological error patterns, in: 17th Annual Conference of the International Speech Communication Association (INTERSPEECH 2016), 2016, pp. 2661–2665.

[40] A. Duenser, L. Ward, A. Stefani, D. Smith, J. Freyne, A. Morgan, B. Dodd, Feasibility of technology enabled speech disorder screening, in: Digital

Health Innovation for Consumers, Clinicians, Connectivity and Community: Selected Papers from the 24th Australian National Health Informatics Conference (HIC 2016), Vol. 227, IOS Press, 2016, p. 21.

[41] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, et al., The Kaldi speech recognition toolkit, in: IEEE 2011 workshop on automatic speech recognition and understanding, no. EPFL-CONF-192584, IEEE Signal Processing Society, 2011.

[42] M. Russell, The PF-STAR british english childrens speech corpus, The Speech Ark Limited, 2006.

[43] P. Howell, L. Vause, Acoustic analysis and perception of vowels in stuttered speech, The Journal of the Acoustical Society of America 79 (5) (1986) 1571–1579.

[44] J. G. Sheehan, Stuttering behavior: A phonetic analysis, Journal of Communication Disorders 7 (3) (1974) 193–212.

[45] S. Alharbi, A. J. Simons, S. Brumfitt, P. Green, Automatic recognition of childrens read speech for stuttering application, in: Proc. WOCCI 2017: 6th International Workshop on Child Computer Interaction, 2017, pp. 1–6.

[46] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, software available from tensorflow.org (2015).
URL `https://www.tensorflow.org/`

[47] A. Kramida, Yu. Ralchenko, J. Reader, and NIST ASD Team, NIST Atomic Spectra Database (ver. 1.5), [Online]. Available: http://www1.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm [2018, Jan 18]. National Institute of Standards and Technology, Gaithersburg, MD. (2018).

Conflict of Interest

- There is no conflict of interest between the reviewers and the submitted paper.
- We are submitting this empty file as a requirement for completing the upload of the corrections requested by the second reviewer.

Best Regards