Re-framing Incremental Deep Language Models for Dialogue Processing with Multi-task Learning

Morteza Rohanian, Julian Hough
Cognitive Science Group
School of Electronic Engineering and Computer Science
Queen Mary University of London
{m.rohanian, j.hough} @qmul.ac.uk

Abstract

We present a multi-task learning framework to enable the training of one universal incremental dialogue processing model with four tasks of disfluency detection, language modelling, part-of-speech tagging and utterance segmentation in a simple deep recurrent setting. We show that these tasks provide positive inductive biases to each other with optimal contribution of each one relying on the severity of the noise from the task. Our live multi-task model outperforms similar individual tasks, delivers competitive performance and is beneficial for future use in conversational agents in psychiatric treatment.

1 Introduction

Conversational technologies offer a remarkable addition to the current approaches for providing mental healthcare. Communications with these conversational agents have been found to include discoverable psychological distress signs, such as the rate of filled pauses, speech rate, and various temporal and turn-related characteristics (Gratch et al., 2014). In human-human automatic analysis of patient-doctor conversations it has also been found that different types of disfluency can indicate levels of adherence to medication (Howes et al., 2012). Markers of disfluency also hold predictive power for identification of cognitive disorders (Rohanian et al., 2020).

Such devices are mainly used for processing content which is then analyzed offline. There is much work on detecting disfluencies for offline analysis of transcripts with gold standard utterance segmentation within much of the current effort on disfluency detection on telephone conversations begun by Charniak and Johnson (2001). However, given that these models do not operate for live systems and rely on rich transcription data including the pre-segmentation of dialogue acts, to facilitate more cost-effective study of other data, it would be easier to be able to perform directly and incrementally off the speech signal, or at least from automatic speech recognition (ASR) results as they arrive into the system. The incremental model must work with minimum latency as it receives word-by-word data and does so without modifying its initial assumptions and providing its best decisions as soon as possible in line with the principles set out in (Hough and Purver, 2014).

We combine incremental identification of disfluencies with three other tasks that are essential for active conversational models to provide a favorable inductive bias to disfluency detection and to study the way these tasks interact. We explore a multi-task learning (MTL) framework to enable the training of one universal model with four tasks of disfluency detection, language modelling, part-of-speech (POS) tagging and utterance segmentation, which in the data we use is also equivalent to dialogue act segmentation. Multi-task learning seeks to improve learning efficiency and predictive power by learning from a shared representation with multiple objectives. We investigate the entire power set of these tasks to investigate the interaction between them. We experiment with two different methods for losses: a naive weighted sum of losses where the weights of loss are uniform, and a loss function based on maximizing the Gaussian likelihood with task-dependent uncertainty.

We train and test a simple neural model for the different tasks, experimenting with all the combinations of the tasks, different loss functions for each of the tasks, and also experiment with different input representations (words vs. words with word duration).

2 Related Work

Although significant research has been done on disfluency detection as an individual task, most of this work uses transcripts as texts rather than using data from live speech, with the intention of 'cleaning' such texts of disfluent content for eventual post-processing. They are performed on pre-segmented utterances in the Switchboard corpus of telephone conversations (Godfrey et al., 1992). Sequence tagging models with start-inside-outside (BIO) style tags have been used in many studies to detect disfluencies. The most common techniques have utilized discriminative models as a classifier like Conditional Random Fields (CRFs) (Lafferty et al., 2001).

Such methods are insufficient if we intend to measure context from repairs and edit words for disfluency detection, that is not only useful in psychiatric domain but also logical for a dialogue framework that aims to measure a clear understanding of user statements.

Methods focusing on incremental performance have been uncommon. Hough and Purver (2014) used a pipeline of classifiers and language model features in a highly incrementally operating system without looking-ahead. Incremental dependency parsing paired with removal of disfluency was also investigated (Rasooli and Tetreault, 2015). Two studies have applied recurrent neural networks for live disfluency detection. One approach, using a simple Elman Recurrent Neural Network (RNN), investigates incremental detection, with an objective coupling detection efficiency with low latency (Hough and Schlangen, 2015). A development of this work proposed a joint task of incremental disfluency detection and utterance segmentation, with the performance of both tasks jointly improving over equivalent systems doing the individual tasks (Hough and Schlangen, 2017).

There has been research on utterance segmentation as an individual task and also in joint models. Cuendet et al. (2006) makes use of a range of lexical and acoustic features. Xu et al. (2014) used prosodic and lexical features to implement a DNN combined with a CRF classifier for broadcast news speech. Atterer et al. (2008) used syntactic ground-truth information, in a rare incremental approach, to predict whether the current word on Switchboard is the end of the utterance (dialog act). Seeker (2016) used utterance segmentation in a joint framework with dependency parsing.

Language models have been used as an auxiliary task for disfluency detection, based on the intuition that disfluency occurrences will be indicated and edited from the context to improve the prediction of the next words (Johnson and Charniak, 2004), most recently using variants of RNN langauge models (Shalyminov et al., 2018). POS tags have also been used as an input feature in detecting disfluencies, with slight boosts in disfluency detection possible (Hough and Schlangen, 2015; Hough and Schlangen, 2017).

In this paper we define a live setting of joint tasks which include disfluency detection, utterance segmentation, language modelling and POS tagging. We present a simple deep learning system after defining the tasks in the next section, which simultaneously detects disfluencies, assigns POS tags, predicts upcoming utterance boundaries and following words from incremental word hypotheses and derived information.

3 The Tasks

Incremental disfluency detection Disfluencies are typically assumed to have a reparandum-interregnum-repair structure, in their fullest form as speech repairs (Shriberg, 1994; Meteer et al., 1995). A reparandum is a stretch of speech subsequently fixed by the speaker; the corrected expression is a repair, the start of which we will refer to as the *repair onset*. An interregnum word is a filler or a reference expression between the words of repair and reparandum, often a halting step as the speaker produces the repair, giving the structure as in (1)

$$\texttt{John}\ \underbrace{\texttt{[likes}}_{\texttt{reparandum}} + \underbrace{\{\ \texttt{uh}\ \}}_{\texttt{interregnum}} \underbrace{\texttt{loves]}}_{\texttt{repair}}\ \texttt{Mary} \tag{1}$$

	| A	uh	flight	[to	Boston +	{ uh	I	mean }	to	Denver]	on	Friday |	Thank	you |
Disfluency	f	e	f	f	f	e	e	e	$rpS{-}5$	$rpnSub$	f	f	f	f
Utterance segmentation	$.w$-	-w-	-w-	-w-	-w-	-w-	-w-	-w-	-w-	-w-	-w-	-w.	$.w$-	-w.
POS tags	DT	UH	NN	IN	NNP	UH	PRP	VB	IN	NNP	IN	NNP	VB	PRP

Figure 1: An utterance with the disfluency tags (repair onsets and edit terms), utterance segmentation annotation tags and POS tags in our incremental tag schemes

In the absence of reparandum and repair, the disfluency reduces to an isolated *edit term*. A marked, lexicalized edit term such as a filled pause ("uh" or "um") or more phrasal terms like "I mean" and "you know" can occur. Recognizing these elements and their structure is then the task of disfluency detection.

The task of detecting incremental disfluencies adds to the challenge of doing this in real time, word-by-word from left to right. Disfluency identification is then cast as the same challenge a human processor faces with a disfluent utterance: only when the interregnum is detected, or perhaps even when the repair onset is encountered, does it become apparent the earlier content is now to be regarded as "to be repaired", i.e. identified as the reparandum. Therefore, the task cannot be established as a simple sequence labeling task in which the tags for the reparandum, interregnum and repair phases are allocated left-to-right over words as seen in the above example; in this case, it will require the assumption that "likes" would be repaired, at a point where there is no evidence to make it available.

We follow Hough and Schlangen (2015), using a tag set that encodes the start of the reparandum only at a time when it can be inferred, mainly when the repair begins – the individual disfluency detection task is to tag words as in the top line of tags in Fig. 1 as either fluent (f) an edit term (e), a repair onset word ($rpS{-}N$ for the reparandum starting N words back) and a repair end word of the type repeat ($rpnRep$), substitution ($rpnSub$) or delete ($rpnDel$).

Incremental utterance segmentation Utterance segmentation has a strong interdependence with disfluency detection, as standard disfluency detection operates on an utterance or dialogue act level. The two tasks may be more difficult done separately as without utterance segmentation, disfluent restarts and repairs may be predicted at fluent utterance boundaries (Hough and Schlangen, 2017), while without disfluency detection the utterance segmentation may predict an utterance boundary at a repair onset point. So we presume that the tasks can be carried out together in a supplementary manner.

We characterize incremental utterance segmentation as the real-time word-by-word decision as to whether the current utterance is ending. We are shifting from strictly reactive, silent-signaled approach to prediction. Following Hough and Schlangen (2017), we use four tags to define ranges of acoustic data which are the time spans of forced aligned gold standard words, equivalent to a BIES utterance scheme (Beginning, Inside, End and Single) to allow for prediction. The tag set allows information to be captured from the previous context of the word to determine whether this word continues an existing utterance (the - prefix) or begins anew (the . prefix), and also allows online prediction of whether the following word will continue the existing utterance (the - suffix) or whether the current word completes the utterance (the . suffix)- see Fig. 1 for the incremental utterance segmentation tags around a w symbol for the current word. Unlike Hough and Schlangen (2017) we do not employ a joint tag set, instead looking to use multi-task learning as described below to combine these tasks into one learning regime.

Incremental POS tagging Part-of-speech (POS) tags can improve disfluency detection on different settings. We combine POS tagging as an additional task with a similar structure to utterance segmentation and disfluency detection to motivate the model to learn better features for syntactic and semantic structures which can improve the other tasks without requiring additional training data. POS tagging itself could be helped by information about utterance segmentation and disfluency structure, as the parallelism between the reparadum and repair in substitutions as shown in the repeated $IN\ NNP$ sequences in Fig. 1 show, allowing better disambiguation of 'to' with this extra information.

Language modelling The central idea of our approach to language modelling is that the probability of the current word can be more accurately modelled with knowledge from the other dialogue phenomena detected as described above, compared to the standard approach of just using the previous word values, modelling it as part of a joint task (Heeman and Allen, 1999). A secondary assumption is that disfluen-

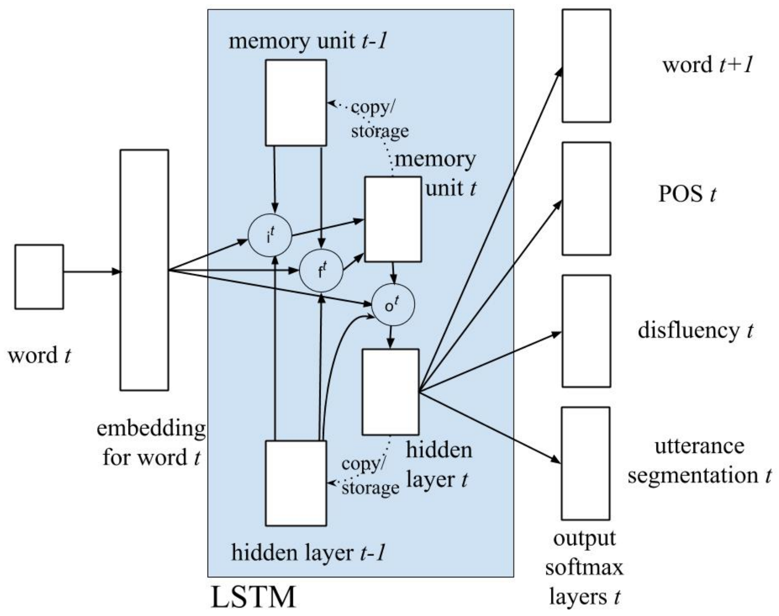

Figure 2: Incremental Long Short-Term Memory net (LSTM) for disfluency detection, utterance segmentation, POS-tagging tasks and language modelling.

cies, each of which has a context-conditioned probability, can be represented as word-like occurrences themselves (Stolcke and Shriberg, 1996). Utterance segmentation and disfluency detection are optimized only on the basis of accurate labels. While each token in the input has a target tag, some contribute next to nothing to the training process. The disfluency detection and utterance segmentation models can acquire a bias in label distribution by gaining additional information from the majority labels. So we are using another objective that would allow the models to make full use of the dataset.

In addition to learning how to predict labels for each word, we also suggest optimizing different parts of the architecture as a language model. The role of predicting the next word would require the model to study more general patterns of semantic and syntactic structure, which can then be reproduced to predict individual words more accurately. This purpose is also generalizable to every task and dataset that does not include additional annotated training details. With a simple adjustment in the model, we introduce a second parallel output layer for each token, improving it to predict the next word. We predict the next word in the sequence only based on the hidden representation from the left-to-right moving LSTM.

3.1 Combining the tasks with multi-task learning

In this paper, following the intuition of Heeman and Allen (1999) that combining these tasks together will improve accuracy of all of them, given their mutual dependence, we take a multi-task learning (MTL) deep learning setting, where we have several prediction tasks over the same input space. Disfluency detection, utterance segmentation, language modelling and POS-tagging are all done simultaneously at each input step. The method can learn how to efficiently adjust the tasks weightings, resulting in a better performance compared with individual learning of each task. Key to optimization is the different type of loss function used for each task, which we investigate in our experiments below.

4 Model architecture

For this task, we use the incremental neural network model the Long Short-Term Memory net (LSTM) as the baseline architecture. For our individual disfluency detection, utterance segmentation and POS-tagging tasks, the model takes the input and uses an LSTM to add a label to each token. For our language modelling task, the model predicts the likelihood of the current word given the previous words. The overall architecture of our model is shown in Figure 2. First, the input tokens are mapped to distributed word embeddings. An LSTM is used for constructing context-dependent representations for every word

(Rei, 2017). The LSTM, together with the word embedding from the current step, takes the hidden state from the preceding time step as input and produces a new hidden state:

$$h_t = LSTM(x_t, h_{t-1}) \tag{2}$$

Then, the representation is passed through a feedforward layer:

$$d_t = tanh(W_d h_t) \tag{3}$$

where $tanh$ is a non-linear activation function and W_d is weight matrix.

We use a Conditional Random Field (CRF) output architecture to predict a tag for each token. Although this model creates predictions according to all the words in the data, the labels are predicted individually. There are high dependencies for disfluency detection between subsequent labels and explicit modeling of those connections can be helpful. The output can be modified to have a CRF which enables the model to test for the most optimal path across all available label sequences. The model is then improved by maximizing the score for the correct label sequence, while reducing the score for all other sequences:

$$E = -s(y) + log \sum_{\widetilde{y} \in \widetilde{Y}} e^{s(\widetilde{y})} \tag{4}$$

where $s(y)$ is the score for the sequence y and Y is the set of all tag sequences.

For our language modelling task, first, using a non-linear layer, the hidden representations from LSTMs are mapped to a new space:

$$m_t = tanh(W_m h_t) \tag{5}$$

where W_m is weight matrix. This distinct transformation learns to obtain features that are specific to language modeling. Then, these representations are carried through softmax layers to predict the likelihood of the current word w_t given the previous words characterized by m_t:

$$P(w_t \mid m_t) = softmax(W_q m_{t-1}) \tag{6}$$

The objective function is an objective of regular language modeling E_l, which is the negative log-likelihood of the current word in the sequence given the previous words.

$$E_l = -log(P(w_t \mid m_t)) \tag{7}$$

The network is designed to predict the current label at each token position in terms of disfluency detection, utterance segmentation and POS tagging and the likelihood of the next word in sequence by predicting the likelihood of each word in the vocabulary. The added language modeling objective enables the model to learn the better representations of words, that are then used for other tasks. To boost MTL's performance on the target dataset and to learn several tasks simultaneously, we propose two loss function methods: first, combining multi-objective losses with a naive approach and simply carrying out a weighted linear loss sum for each task:

$$\widetilde{E} = E + \alpha(E_l) \tag{8}$$

α is the parameter that we use to control the importance of auxiliary tasks (language modelling here). Secondly, inspired by (Kendall et al., 2018) we use an orthogonal approach, taking the uncertainty of each task into account. This multi-task loss learn various classification and regression losses of varying quantities and units simultaneously using task uncertainty. We adjust the relative weight of each task in cost function by deriving a multi-task loss function based on maximizing the Gaussian likelihood with task-dependent uncertainty, generalizing this to our four tasks each with their separate σ weight:

$$\widetilde{E} = \sum_{i=1}^{4} \frac{E_i}{\sigma_i^2} + \sum_{i=1}^{4} log(\sigma_i) \tag{9}$$

with σ the model's trainable observation noise parameter – capturing how much noise we have in the outputs. E_i is one of four indexed to the individual tasks: E_{LM}, E_{Disf}, E_{seg} and E_{POS}.

5 Experimental Set-up

With various tasks, we assess the proposed architecture in joint models with different number of tasks. We experimented with two types of inputs: word embeddings and the duration of the current word in addition to the word embeddings. The word embedding was initialized with pre-trained vectors available to the public, generated using 50-dimensional embedding trained on Google News (Mikolov et al., 2013). The neural network model has been implemented using Tensorflow (Abadi et al., 2016). The LSTM hidden layers are 200. To minimize computational complexity in these experiments, the language modeling task predicted only the 7000 most commonly used words, with an additional token representing all the other words. Results on the development set was also used to find the best model to be evaluated on the test set. We train all models for a maximum of 50 epochs, otherwise stop training if there is no advancement on the best score on the transcript validation set after 7 epochs. The code used in the experiments are publicly available in an online repository.[1]

Data We use standard Switchboard training data (all conversation numbers startning sw2*,sw3 * in the Penn Treebank III release: 100k utterances, 650 K words) and use standard heldout data (PTB III files sw4[5-9] *: 6.4 K utterances, 49 K words) as our validation set. We test standard test data (PTB III files 4[0-1] *) with partial words and punctuation stripped away from all files.

Evaluation Criteria We calculate F1 accuracy for repair onset detection F_{rpS} and for edit term words F_e, which includes interregna. For utterance segmentation we also use word-level F1 scores for utterance boundaries (end-of- utterance words) F_{uttSeg}. We calculate accuracy ACC_{POS} for POS tagging and $Perplexity$ for the language modelling task.

We measure latency and also the stability of output over time, which is essential to the live nature of the model. We use the first time to detection (FTD) metric of (Zwarts et al., 2010) for latency: the average distance (in number of words) consumed before the first detection of gold standard repairs from the repairs onset word. For stability, from the evaluation of incremental processors by (Baumann et al., 2011), we evaluate the edit overhead (EO) of the output labels – the proportion of the unnecessary editing (insertion and deletion) necessary to attain the final labels produced by the model.

6 Results

Our best utterance-final accuracies from different tasks with a loss function based on uncertainty are shown in Table 1. Our best F_{rpS} reaches 0.743 and best F_e reaches 0.922. For utterance segmentation, F_{uttSeg} reaches 0.767. It's difficult to compare our results with the standard approaches for the detection of disfluency as they use pre-segmented utterances. However our best result is as high as (Seeker, 2016)'s models on the Switchboard data for utterance segmentation that is state-of-the-art. In comparison to incremental approaches, we outperform Hough and Schlangen (2017)'s 0.748 on end-of-utterance and Hough and Schlangen (2015) and Hough and Schlangen (2017)'s 0.720 and 0.918 on F_{rpS} and F_e.

The models using the timing outperform those with lexical information only on the utterance segmentation metrics and F_e, whilst having lower performance on F_{rpS}, language modelling and POS tagging. We get our highest results in all tasks from the model with four joint tasks. Edit term detection works very well at 0.922, nearing the state-of-the-art on the Switchboard reported at 0.938. Our lowest perplexity is 64.3. Additional training objective leads to more accurate language modelling in all joint models except when it is trained with disfluency detection alone. The best result for POS tagging is 0.965. It is important to mention that it's difficult to do LM and POS tagging comparisons on this particular dataset. However the high quality of our neural POS tagging and language modelling on un-segmented data and

[1]https://github.com/mortezaro/mtl-disfluency-detection

Table 1: Non-incremental (dialogue-final) results for different tasks with a loss function based on uncertainty of tasks

Input	Models	F_{uttSeg}	F_e	F_{rpS}	ACC_{POS}	$Perplexity$
Baselines						
Words	Seeker et al. (2016)	0.767	-	-	-	-
Words	Hough and Schlangen (2015)	-	0.902	0.689	-	-
Words / Words + Timings	Hough and Schlangen (2017)	0.748	0.918	0.720	-	-
Our Models						
Words	Single Tasks	0.689	0.904	0.678	0.961	65.3
	LM + uttSeg	0.725	-	-	-	65.0
	LM + Disf	-	0.915	0.717	-	67.5
	POS + Disf	-	0.913	0.713	0.961	-
	uttSeg + Disf	0.709	0.917	0.718	-	-
	uttSeg + Disf + LM	0.734	0.917	0.724	-	**64.3**
	uttSeg + Disf + LM + POS	0.763	0.917	**0.743**	**0.965**	**64.3**
Words + Timings	LM + uttSeg	0.728	-	-	-	65.2
	LM + Disf	-	0.915	0.711	-	67.5
	POS + Disf	-	0.913	0.712	0.960	-
	uttSeg + Disf	0.712	0.917	0.715	-	-
	uttSeg + Disf + LM	0.738	0.919	0.720	-	64.4
	uttSeg + Disf + LM + POS	**0.767**	**0.922**	0.741	0.964	64.5

in a live setting are comparable to state-of-the-art approaches, and are useful enough to improve utterance segmentation and disfluency detection, in line with our main goal.

Joint tasks As can be seen in the Table 1, the overall highest scoring systems for individual tasks do not achieve results in any relevant metric of the top performing joint systems. We achieve the highest F_{rpS}, F_e, F_{uttSeg}, ACC_{POS} and $Perplexity$ with a model with all four tasks. Adding language modelling, utterance segmentation and POS tagging also helps disfluency detection in models with two and three joint tasks. While the performance improvements are small, they are consistent across all joint models. Joint models with more than two tasks does not have an impact on F_e when the input is only lexical. Our joint model with four tasks outperforms Hough and Schlangen (2017)'s joint model in both disfluency detection and utterance segmentation.

F_{uttSeg} is the metric with the most improvements with the addition of relevant tasks as it gets higher in all the joint model. Language modelling helps utterance segmentation more than disfluency detection in joint models with two tasks. POS tagging does not help utterance segmentation and disfluency detection as much as language modelling in joint models with two tasks but improves the results in comparison to single tasks.

We also get the best results for language modelling with the joint model with three and four tasks ($Perplexity$ 64.3). Adding disfluency detection to language modelling on its own actually decreases the performance ($Perplexity$ 65.3 vs. 67.5). Utterance segmentation is the single most beneficial auxiliary task for language modelling ($Perplexity$ 65.0).

The baseline performance in POS tagging is close to the upper bound, therefore the language modelling, disfluency detection and utterance segmentation objectives do not provide much added advantage.

Timing as input Adding timing information as input provides a consistent improvement across all joint tasks in comparison to single tasks as we can observe in the Table 1. Timing improves utterance segmentation performance in models with two tasks more than other tasks. Timing can help to obtain more information for detecting edit terms in joint tasks. The largest benefit from the timing was observed on the joint task of utterance segmentation and edit term detection. Language modelling performance decreases in all joint models with word timing compared to models without it except in the joint model with disfluency detection.

Utility of loss function using uncertainty From Table 2 we observe that our 'naive' loss function baseline simply using the sum of all the separate losses leads to decrease of performance in all tasks

Table 2: Non-incremental (dialogue-final) results for different tasks with a naive loss function

Models	F_{uttSeg}	F_e	F_{rpS}	ACC_{POS}	$Perplexity$
Single Tasks	0.689	0.904	0.678	0.961	65.3
LM + uttSeg	0.691	-	-	-	64.8
LM + Disf	-	0.903	0.687	-	65.5
POS + Disf	0.665	0.904	0.684	0.957	-
uttSeg + Disf	0.683	0.907	0.697	-	-
uttSeg + Disf + LM	0.667	0.903	0.682	-	64.5
uttSeg + Disf + LM + POS	0.662	0.902	0.690	0.956	64.5

in joint models compared to the uncertainty loss functions except language modelling. Utterance segmentation gets the most unfavorable results with naive loss function compared to other tasks. Adding disfluency detection decreases the utterance segmentation performance the most when it is an auxiliary task. Language modelling is the best auxiliary task across different joint models and gets the best results compared to the same models with loss functions with uncertainty. Using naive loss functions in some cases does not lead to improvement in comparison to the single task models. Single-task utterance segmentation outperforms all the joint models with naive loss except the joint model with language modelling in F_{uttSeg}. Single-task disfluency detection gets better F_e than all the joint models with naive loss except the joint model with POS tagging. Single POS tagging gets better ACC_{POS} than all joint models with naive loss. An increasing number of tasks make it more difficult to obtain optimal weights roughly. Using the uncertainty loss function improves all tasks in all models but utterance segmentation and disfluency detection performances improve the most.

We show that performances in multi-task learning settings for all of our tasks are heavily reliant on the suitable choice of loss weightings. We note that the optimal weighting of each task depends on the severity of the noise from the task.

Incrementality As shown in the Table 4, the incremental performance of our individual disfluency task was better overall than the best performance in joint model settings, with low EO in our best setting at 1.08 and FTD just slightly above 1 word (1.01). If we define EO as the proportion of unnecessary edits to get to the final labels, we can see that adding language modelling increases EO significantly and makes the system less stable (2.01). Adding other classification tasks (POS tagging and utterance segmentation) does not decrease the incremental performance as much as language modelling.

In terms of FTD, all of our models are very fast and close to 1. Our individual disfluency model gets the best score. Adding language modelling slightly decreases the incremental performance for FTD (1.09 vs. 1.01). Joint disfluency detection models with utterance segmentation, POS tagging and language modelling incremental performance are still comparable to individual disfluency detection model in FTD performance. Overall, incremental disfluency tagging performance is decreased most by adding language modelling as a joint task, but this comes with the trade-off of a better final accuracy.

7 Error Analysis

Different repair types Categorizing repetitions as verbatim repeats, substitutes as the other repairs marked with a repair phase, and deletes as those without one, we see in Table 5 that our joint models get better results than the separate disfluency detection task on repeat and substitution types. Separate disfluency detection outperforms joint models in detecting deletes. It seems that adding other tasks to joint models does not improve the accuracy of detecting the rare delete repairs, but helps the accuracy of substitution repairs.

Adding other tasks to the task of disfluency detection improves the repetitions detection but we don't observe an improvement in models with more than two tasks. While the performance improvements of joint models in detecting substitutions are small, they are consistent across all models with different number of tasks. We get the best result for substitutions in our single disfluency model at 0.65 and for repeats in our joint model with language modelling at 0.93 when the loss function is naive. For deletes, single disfluency detection model and joint segmentation and disfluency detection models get the best results at 0.34.

Table 3: F_{rpS} repairs with different reparandum lengths

Models	1	2	3	4	5
Disf	.84	.67	.40	.31	.13
LM + Disf	.84	.68	.42	.33	.13
POS + Disf	.84	.67	.41	.31	.13
uttSeg + Disf	.84	.67	.40	.30	.13
uttSeg + Disf + LM	.84	.68	.43	.33	.13
uttSeg + Disf + LM + POS	.85	.68	.43	.33	.13

Table 4: Incremental results for disfluency detection

Models	FTD	EO
Disf	1.01	1.08
LM + Disf	1.09	2.01
POS + Disf	1.03	1.14
uttSeg + Disf	1.02	1.11
uttSeg + Disf + LM	1.10	2.05
uttSeg + Disf + LM + POS	1.12	2.04

Table 5: $F1$ for repairs with different types

	Uncertainty-based loss			Naive loss		
Models	Repeats	Substitution	Deletes	Repeats	Substitution	Deletes
Disf	.94	.70	.48	.91	.65	.34
LM + Disf	.96	.71	.46	.93	.63	.32
POS + Disf	.96	.70	.47	.90	.63	.33
uttSeg + Disf	.96	.70	.46	.90	.64	.34
uttSeg + Disf + LM	.96	.71	.46	.91	.63	.31
uttSeg + Disf + LM + POS	.96	.72	.46	.91	.63	.30

Different repair lengths While our best system still suffers from a vanishing gradient problem to predict repairs with longer reparanda, we can observe in Table 3 an F1 boost for repair lengths 1 to 4 in our joint models. Our joint model with four task gets the best results with repair length 1 to 4.

Our joint models do not show improvements in detecting longer repairs with reparanda of 5 words or longer. The combined number of instances in the test set with reparanda over 5 words are 59 and our best model predict 26 of them correctly.

8 Conclusion

We have presented a multi-task learning framework to enable the training of one universal incremental model with four tasks of disfluency detection, language modelling, part-of-speech tagging and utterance segmentation. We have observed that these tasks produce favorable inductive biases to each other with utterance segmentation and disfluency detection get the most benefits. We note that the optimal weighting of each task relies heavily on the severity of the noise from the task. We showed that word timing information helps the utterance segmentation and disfluency detection in online setting and adding new tasks with the exception of language modelling does not have a remarkable negative effect on the incremental metrics.

The results show that our framework can be suitable for online conversational systems, such as conversational agents in the mental health domain. In future work we intend to analyze the interactions between different task as they occur in real time. Monitoring the interaction after each word could help highlight informative moments that contribute more to optimisation of our models. Furthermore, we intend to use raw acoustic features as the input for a strongly time-linear model.

Acknowledgments

We thank the anonymous COLING reviewers for their helpful comments and Matthew Purver for his continuous support and supervision on the wider project.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283.

Michaela Atterer, Timo Baumann, and David Schlangen. 2008. Towards incremental end-of-utterance detection in dialogue systems. In *Proceedings of the 22nd International Conference on Computational Linguistics*.

Timo Baumann, Okko Buß, and David Schlangen. 2011. Evaluation and optimisation of incremental processors. *Dialogue & Discourse*, 2(1):113–141.

Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Sébastien Cuendet, Dilek Hakkani-Tur, and Gokhan Tur. 2006. Model adaptation for sentence segmentation from speech. In *2006 IEEE Spoken Language Technology Workshop*, pages 102–105. IEEE.

John J Godfrey, Edward C Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, volume 1, pages 517–520. IEEE Computer Society.

Jonathan Gratch, Ron Artstein, Gale M Lucas, Giota Stratou, Stefan Scherer, Angela Nazarian, Rachel Wood, Jill Boberg, David DeVault, Stacy Marsella, et al. 2014. The distress analysis interview corpus of human and computer interviews. In *LREC*, pages 3123–3128.

Peter A Heeman and James Allen. 1999. Speech repains, intonational phrases, and discourse markers: modeling speakers' utterances in spoken dialogue. *Computational Linguistics*, 25(4):527–572.

Julian Hough and Matthew Purver. 2014. Strongly incremental repair detection. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 78–89.

Julian Hough and David Schlangen. 2015. Recurrent neural networks for incremental disfluency detection. In *Sixteenth Annual Conference of the International Speech Communication Association*.

Julian Hough and David Schlangen. 2017. Joint, incremental disfluency detection and utterance segmentation from speech. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 326–336.

Christine Howes, Matt Purver, Rose McCabe, Patrick GT Healey, and Mary Lavelle. 2012. Helping the medicine go down: Repair and adherence in patient-clinician dialogues. In *Proceedings of SemDial 2012 (SeineDial): The 16th Workshop on the Semantics and Pragmatics of Dialogue*, page 155.

Mark Johnson and Eugene Charniak. 2004. A TAG-based noisy-channel model of speech repairs. In *ACL*, pages 33–39.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

M. Meteer, A. Taylor, R. MacIntyre, and R. Iyer. 1995. Disfluency annotation stylebook for the switchboard corpus. ms. Technical report, Department of Computer and Information Science, University of Pennsylvania.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. version 2.

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. *arXiv preprint arXiv:1704.07156*.

Morteza Rohanian, Julian Hough, and Matthew Purver. 2020. Multi-Modal Fusion with Gating Using Audio, Lexical and Disfluency Features for Alzheimer's Dementia Recognition from Spontaneous Speech. In *Proc. Interspeech 2020*, pages 2187–2191.

Anders Seeker. 2016. How to train dependency parsers with inexact search for joint sentence boundary detection and parsing of entire documents.

Igor Shalyminov, Arash Eshghi, and Oliver Lemon. 2018. Multi-task learning for domain-general spoken disfluency detection in dialogue systems. In *Proceedings of the 22nd SemDial Workshop on the Semantics and Pragmatics of Dialogue (AixDial)*, Aix-en-Provence, November.

Elizabeth Shriberg. 1994. *Preliminaries to a Theory of Speech Disfluencies*. Ph.D. thesis, University of California, Berkeley.

Andreas Stolcke and Elizabeth Shriberg. 1996. Statistical language modeling for speech disfluencies. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 1, pages 405–408. IEEE.

Chenglin Xu, Lei Xie, Guangpu Huang, Xiong Xiao, Eng Siong Chng, and Haizhou Li. 2014. A deep neural network approach for sentence boundary detection in broadcast news. In *Fifteenth annual conference of the international speech communication association*.

Simon Zwarts, Mark Johnson, and Robert Dale. 2010. Detecting speech repairs incrementally using a noisy channel approach. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1371–1378.