Using phone features to improve dialogue state tracking generalisation to unseen states

Iñigo Casanueva, Thomas Hain, Mauro Nicolao, and Phil Green
Department of Computer Science, University of Sheffield, United Kingdom
{i.casanueva, t.hain, m.nicolao, p.green}@sheffield.ac.uk

Abstract

The generalisation of dialogue state tracking to unseen dialogue states can be very challenging. In a slot-based dialogue system, dialogue states lie in discrete space where distances between states cannot be computed. Therefore, the model parameters to track states unseen in the training data can only be estimated from more general statistics, under the assumption that every dialogue state will have the same underlying state tracking behaviour. However, this assumption is not valid. For example, two values, whose associated concepts have different ASR accuracy, may have different state tracking performance. Therefore, if the ASR performance of the concepts related to each value can be estimated, such estimates can be used as general features. The features will help to relate unseen dialogue states to states seen in the training data with similar ASR performance. Furthermore, if two phonetically similar concepts have similar ASR performance, the features extracted from the phonetic structure of the concepts can be used to improve generalisation. In this paper, ASR and phonetic structure-related features are used to improve the dialogue state tracking generalisation to unseen states of an environmental control system developed for dysarthric speakers.

1 Introduction

Dialogue state tracking (DST) (Thomson and Young, 2010) is a key component for spoken interfaces for electronic devices. It maps the dialogue history up to the current dialogue turn (Spoken language understanding (SLU) output, actions taken by the device, etc.) to a probabilistic representation over the set of *dialogue states*[1] called the *belief state* (Young et al., 2013). This representation is the input later used by the dialogue policy to decide the next action to take (Williams and Young, 2007; Gašić and Young, 2014; Geist and Pietquin, 2011). In the Dialogue State Tracking Challenges (DSTC) (Williams et al., 2013; Henderson et al., 2014), it was shown that data driven discriminative models for DST outperform generative models in the context of a slot based dialogue system. However, generalisation to unseen dialogue states (e.g. changing the dialogue domain or extending it) remains an issue. The 3rd DSTC (Henderson et al., 2014b) evaluated state trackers in extended domains, by including dialogue states not seen in the training data in the evaluation data. This challenge showed the difficulty for data-driven approaches to generalise to unseen states, as several machine learned trackers were outperformed by the rule-based baseline. Data driven state trackers with slot-specific models cannot handle unseen states. Therefore, general state trackers track each value independently using general value-specific features (Henderson et al., 2014c; Mrksic et al., 2015). However, dialogue states are by definition in discrete space where similarities cannot be computed. Thus, a general state tracker has to include a general value-tracking model that can combine the statistics of all dialogue states. This strategy assumes that different dialogue states have the same state tracking behaviour, but such assumption is rarely true. For example, two values, whose associated concepts have different ASR accuracy, have differ-

[1] In a slot based dialogue system the dialogue states are defined as the set of possible value combinations for each slot. However, in this paper we use *dialogue states* to refer to the set of *slot-value* pairs and *joint dialogue states* to the actual dialogue states.

Proceedings of the SIGDIAL 2016 Conference, pages 80–89,
Los Angeles, USA, 13-15 September 2016.

ent state tracking performance. A general feature able to define similarities between dialogue states would improve state tracking generalisation to unseen states, as the new values could be tracked using statistics learned from the most similar states seen in the training data.

Dialogue management was shown to improve the performance of spoken control interfaces personalised to dysarthric speakers (Casanueva et al., 2014; Casanueva et al., 2015). For these type of interfaces (e.g. homeService (Christensen et al., 2013; Christensen et al., 2015)), the user interacts with the system using single word commands[2]. Each slot-value in the system has its associated command. It is a reasonable assumption that two dialogue states or values associated to commands with similar ASR accuracy will also have similar DST performance. If the ASR performance of commands can be estimated (e.g. in a held out set of recordings), the measure can be used as a general feature to help the state tracker relate unseen dialogue states to similar states seen in the training data.

However, a held out set of recordings can be costly to obtain. If it is assumed that phonetically similar commands will have similar recognition rates, general features extracted from the phonetic structure of the commands can be used. For example, the ASR can find "problematic phones", i.e. phones or phone sequences that are consistently misrecognised. Therefore, the state tracker can learn to detect such problematic phones and adapt its dialogue state inference to the presence of these phones. If an unseen dialogue state that contains these phone patterns is tracked, the state tracker can infer the probability of that state more efficiently. Using the command phonetic structure as additional feature for state tracking can be interpreted as moving from state tracking in the "command space", where similarities between dialogue states cannot be computed, to state tracking in the "phone space", where those similarities can be estimated.

In this paper, we propose a method to use ASR and phone-related general features to improve the generalisation of a Recurrent Neural Network (RNN) based dialogue state tracker to unseen states. In the next section, state-of-the-art methods for generalised state tracking are described. Following section describes the proposed ASR and phone-related features as well as different approaches to encode variable length phone sequences into fixed length vectors. Section 4 describes the experimental set-up. Sections 5 and 6 present results and conclusions.

[2]Severe dysarthric speakers cannot articulate complete sentences.

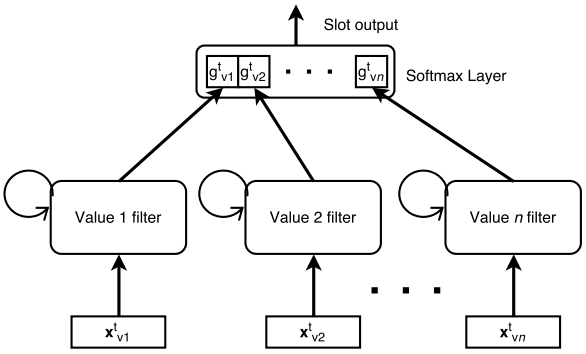

Figure 1: *General DST for a single slot.*

2 Generalised dialogue state tracking

In slot-based dialogue state tracking, the *ontology* defines the set of slots $\mathcal{S}$ and the set of possible values for each slot $\mathcal{V}_s$. A dialogue state tracker is hence a classifier, where classes correspond to the joint dialogue states. However, slot-based trackers often factorise the joint dialogue state into slots and therefore use a classifier to track each slot independently (Lee, 2013). Then, the set of values for that slot $\mathcal{V}_s$ are the classes. The joint dialogue state is computed by multiplication and renormalisation of individual probabilities for each slot. Even if the factorisation of the dialogue state helps to generalise by reducing the number of effective dialogue states or values to track, slot specifically trained state trackers are not able to generalise to unseen values as they learn the specific statistics of each slot and value. State trackers able to generalise to unseen values track the probability of each value independently using value specific general features, such as the confidence score of the concept associated to that value in the SLU output (Henderson et al., 2014d).

2.1 Rule based state tracking

Rule-based state trackers (Wang and Lemon., 2013; Sun et al., 2014b) use slot-value independent rules to infer the probability of each dialogue state. An example is the sum of confidence scores of the concept related to that value or the answers confirming that the value is correct. Rule based methods show a competitive performance when evaluated in new or extended domains, as

it was demonstrated in the 3rd DSTC. However, low adaptability can reduce the performance in domains that are challenging for ASR.

2.2 Slot-value independent data-driven state tracking

In the first two DSTC, most of the data driven approaches to dialogue state tracking learned specific statistics for each slot and value (Lee, 2013; Williams, 2014). However, in some cases (Lee and Eskenazi., 2013), parameter tying was used across slot models, thereby assuming that the statistics of two slots can be similar. The 3rd DSTC addressed domain extension, and state trackers able to generalise to unseen dialogue states had to be developed. One of the most successful approaches (Henderson et al., 2014d) combined the output of two RNNs trackers: one represented slot-specific statistics and the other modelled slot-value independent general statistics. Later, (Mrksic et al., 2015) modified this model to be able to track the dialogue state in completely different domains by using only the general part of the model of (Henderson et al., 2014d). The slot-value independent model (shown in Fig. 1) comprises of a set of binary classifiers or *value filters*[3], one for each slot-value pair, with parameters shared across all filters. These filters track each value independently, and the slot s output distribution in each turn is obtained by concatenating the outputs of each value filter g_v^t in $\mathcal{V}_s$, followed by applying a softmax function. The set of filters only differs from each other in two aspects: in the input composed by value specific general features (also called *delexicalized features*); and in the label used during the training. An RNN-based general state tracker[4] updates the probability of each value p_v^t in each turn t as follows:

$$\begin{aligned} \mathbf{h}_v^t &= \sigma(\mathbf{W}_x \mathbf{x}_v^t + \mathbf{W}_h \mathbf{h}_v^{t-1} + \mathbf{b}_h) \\ g_v^t &= \sigma(\mathbf{w}_g \mathbf{h}_v^t + b_g) \\ p_v^t &= \frac{\exp(g_v^t)}{\sum_{v' \in V} \exp(g_{v'}^t)} \end{aligned} \quad (1)$$

Where $\mathbf{h}_v^t$ is the hidden state of each filter, $\mathbf{x}_v^t$ are the value specific inputs and $\mathbf{W}_x$, $\mathbf{W}_h$, $\mathbf{b}_h$, $\mathbf{w}_g$ and b_g are the parameters of the model.

[3] Addressed as *filters* due to their resemblance with convolutional neural networks filters.

[4] This is a simplified version of the model described in (Mrksic et al., 2015).

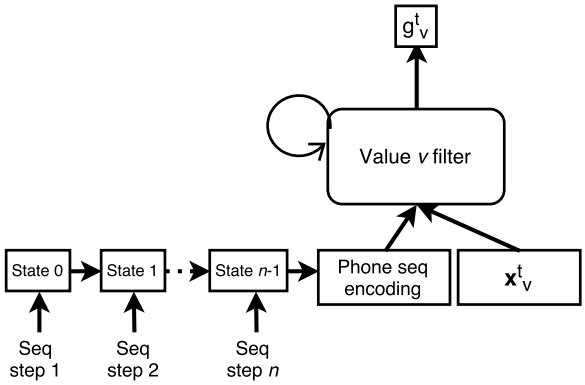

Figure 2: *Joint RNN encoder.*

3 ASR and phone-related general features

The model explained in section 2.2 works with value-specific general features $\mathbf{x}_v^t$ (e.g. the confidence score seen for that particular value in that turn). These features do not help to relate dialogue states with similar state tracking performance, thus the model has to learn the mean statistics from all the states. However, different values have different state tracking performance. Features that can give information about the ASR performance or that can be used to relate the state tracking performance of values seen in the training data to unseen states, should allow to generalise to new dialogue states. In the following section, we introduce various features that can improve generalisation.

3.1 ASR features

In a command-based environmental control system, if recordings of the commands related to the unseen dialogue states are available, they can be used to estimate the ASR performance for the new commands. Then, the value specific features for each filter can be extended by concatenating the ASR accuracy of that specific value. When the tracker faces a value not seen in the training data, it can improve the estimation of the probability of that value by using the statistics learnt form values with similar ASR performance.

3.2 Phone related features

In the previous section, accuracy estimates were proposed to improve general state tracking accuracy. However, these features would have to be inferred from a held out set of word recordings, which may not always be available. In order to avoid this requirement, the phonetic structure of

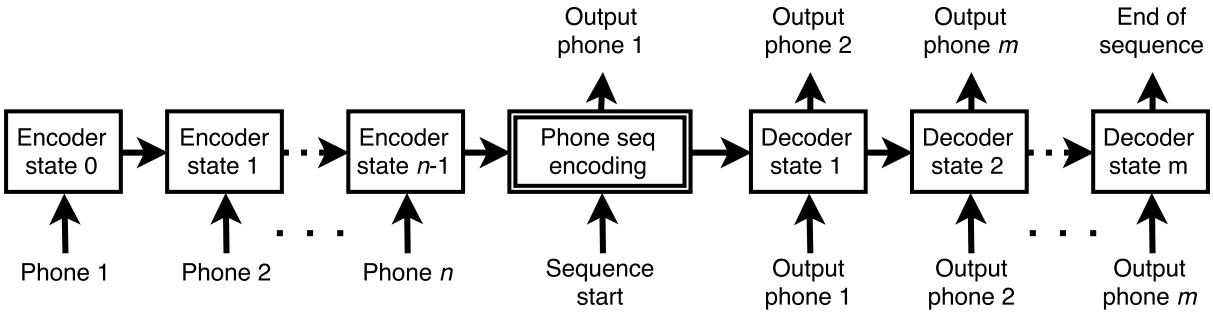

Figure 3: *Seq2seq phone encoder.*

the commands can be used to find similarities between dialogue states with similar ASR performance. The phonetic structure of the commands can be seen as a space composed by subunits of the commands, where similarities between states can be computed.

Phone related features can be extracted in several ways. A deep neural network trained jointly with the ASR can be used to extract a sequence of phone posterior features, one vector per speech frame (Christensen et al., 2013b). Another way is to use a pronunciation dictionary to decompose the output of the ASR into sequences of phones. The later method can be also used to extract a "phonetic fingerprint" of the associated value for each filter. For example, a filter which is tracking the value "RADIO", would have the sequence of phones "r-ey-d-iy-ow" as phonetic fingerprint.

In each dialogue turn, these features are based on sequences of different length. In the case of the ASR phone posteriors, the sequence length is equal to the number of speech frames. When using a pronunciation dictionary, the length is equal to the number of phonemes in the command. However, in each dialogue turn, a fixed length vector should be provided as input of the tracker. Thus, a method to transform these sequences into fixed length vectors is needed. A straightforward method is to compute the mean vector of the sequence, thereby loosing the phone order information. In addition, the number of phones that the sequence has would affect the value of each phone in the mean vector. To compress these sequences in fixed length vectors while maintaining the ordering and the phone length of the sequence, we propose to use a RNN encoder (Cho et al., 2014). We propose two ways to train this encoder, jointly with the model, and with a large pronunciation dictionary.

3.2.1 Joint RNN phone encoder

The state of an RNN is a vector representation of all the previous sequence inputs seen by the model. Therefore, the final state after processing a sequence can be seen as a fixed length encoding of the sequence. If this encoding is put to the filters of the state tracker (Fig. 2), the tracker and the encoder can be trained jointly using back-propagation. We propose to concatenate the encoding of the phonetic sequence in each turn with the value specific features $\mathbf{x}_v^t$ for each filter as shown in Fig. 2. This defines a structure with two stacked RNNs, one encoding the phonetic sequences per turn and the other processing the sequence of dialogue turns.

3.2.2 Seq2seq phone encoder

The need to encode the phone sequences into fixed length "dense" representations which allow to compute similarities, resembles the computing of word embeddings (Mikolov et al., 2013). The difference lies in the fact that word embedding transforms one-hot encodings of words into dense vectors, while in the scope of this work we transform *sequences* of one-hot encodings of phones into dense vectors. Sequence to sequence models (a.k.a. *seq2seq* models, RNN encoder-decoders), can be used to perform such a task. These models consist of two RNNs; an *encoder* which processes the input sequence into a fixed length vector (the final RNN state); and a *decoder*, which "unrolls" the encoded state into an output sequence (Fig. 3). These models have shown state-of-the-art performance in machine translation tasks (Cho et al., 2014), and have been applied to text-based dialogue management with promising results (Lowe et al., 2015; Wen et al., 2016). For the task of generating dense representations of phone sequences, the seq2seq model is trained in a similar way to auto-encoders (Vincent et al., 2008), where in-

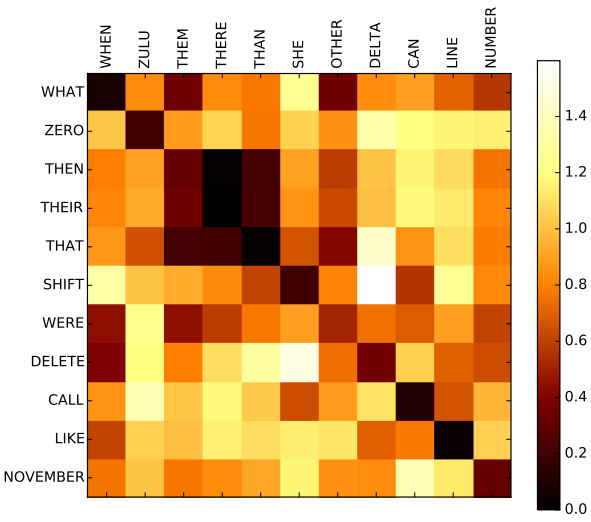

Figure 4: *Cosine distance in the phone encoding space of different words of the UASpeech database.*

put and target sequences are the same, forcing the model to learn to reconstruct the input sequence. The final state of the encoder RNN (the two-line block in Fig. 3) is taken as dense representation of the phone sequence. For this task, the *combilex* pronunciation dictionary (Richmond et al., 2010) is used to train the model. An RNN composed of two layers of 20 LSTM units is able to reconstruct 95% of the phone sequences in an independent evaluation set. This means compressing sequences of one-hot vectors of size 45 (the number of phones in US English) into a vector of size 20. In Fig. 4, the cosine distance between the dense phone representations of two sets of words of the UASpeech database (see sec. 4.1.1) is plotted, illustrating that these encodings are able to effectively relate words with similar phone composition.

4 Experimental setup

The experiments are performed within the context of a voice-enabled control system designed to help speakers with dysarthria to interact with their home devices (Christensen et al., 2013; Casanueva et al., 2016). The user can interact with the system in a mixed initiative way, speaking single-word commands from a total set of 36. As the ASR is configured to recognise single words (Christensen et al., 2012), the SLU operates a direct mapping from the ASR output, an N-Best list of words, to an N-Best list of commands. The dialogue state of the system is factorized into three slots, with the values of the first slot representing the devices to control (TV, light, bluray...), the second slot its functionalities (channel, volume...) and the third slot the actions that these functionalities can perform (up, two, off...). The slots have 4, 17 and 15 values respectively, and the combination of the values of the three slots compose the joint dialogue state or *goal* (e.g. TV-channel-five, bluray-volume-up). The set of valid[5] joint goals $\mathcal{J}$ has a cardinality of 63, and the belief state for each joint goal j is obtained by multiplying the slot probabilities of each of the individual slot values and normalising:

$$P(j) = \frac{P_{s1}(j_1)P_{s2}(j_2)P_{s3}(j_3)}{\sum_{h\in\mathcal{J}} P_{s1}(h_1)P_{s2}(h_2)P_{s3}(h_3)} \quad (2)$$

where $P_{sx}(j_x)$ is the probability of the value j_x in slot s_x and $j = (j_1, j_2, j_3)$.

4.1 Dialogue corpus

One of the main problems in dialogue management research is the lack of annotated dialogue corpora. The corpora released for the first three DSTCs aimed to mitigate this problem. However, this corpus does not include acoustic data. Hence, features extracted from the acoustics such as phone posteriors cannot be used. A large part of dialogue management research relies on *simulated users* (SU) (Georgila et al., 2006; Schatzmann et al., 2007; Thomson et al., 2012) for collection of the data needed. The dialogue corpus used in the following experiments has been generated with simulated users interacting with a rule based dialogue manager. To simulate data collected from dysarthric speakers, a set of 6 SUs with dysarthria has been created.

To simulate data in two different domains, two environmental control systems are simulated, each controlled with a different vocabulary of 36 commands. 72 commands selected from the set of 155 more frequent words in the UASpeech database (Kim et al., 2008), and split into 2 groups, which are named *domain A* and *domain B*. 1000 dialogues are collected in each domain[6]. To be sure that the methods work independently of the set of commands selected, 3 different vocabularies of 72 words are randomly selected and the results presented in the following section show the mean results for the 3 vocabularies.

[5]Many combinations of slot values are not valid sequences, e.g. light-channel-on.

[6]200 extra dialogues are collected in *domain B* for the second set of experiments in section 5.

4.1.1 Simulated dysarthric users

Each SU is composed of a *behaviour simulator* and an *ASR simulator*. The *behaviour simulator* decides on the commands uttered by the SU in each turn. It is rule-based and depending on the machine action, it chooses a command corresponding to the value of a slot or answers a confirmation question. To simulate confusions by the user, it uses a probability of producing a different command, or of providing a value for a different slot than the requested one. The probabilities of confusion vary to simulate different expertise levels with the system. Three different levels are used to generate the corpus to increase its variability.

The *ASR simulator* generates ASR N-best outputs. These N-best lists are sampled from ASR outputs of commands uttered by dysarthric speakers from the UASpeech database, using the ASR model presented in (Christensen et al., 2014). To increase the variability of the data generated, the time scale of each recording is modified to 10% and 20% slower and 10% and 20% faster, generating more ASR outputs to sample from. Phone posterior features are generated as described in (Christensen et al., 2013b) without the principal component analysis (PCA) dimensionality reduction. Six different SUs, corresponding to low- and mid-intelligible speakers, are created from the UASpeech database. ASR accuracy on these users ranges from 32% to 60%.

4.1.2 Rule-based state tracker

One of the trackers used in the DSTCs as baseline (Wang and Lemon., 2013) has been used to collect the corpus. This baseline tracker performed competitively in the 3 DSTCs, proving its capability to generalise to unseen states. The state tracking accuracy of this tracker is also used as the baseline in the following experiments.

4.1.3 Rule-based dialogue policy

The dialogue policy used to collect the corpus follows simple rules to decide the action to take in each turn. For each slot, if the maximum belief of that slot is below a threshold, the system will ask for that slot's value. If the belief is above that threshold but below a second one, it will confirm the value. If the maximum beliefs of all slots are above the second threshold, it will take the action corresponding to the joint goal with the highest probability. The thresholds values are optimised by grid search to maximise the dialogue reward. In addition, the policy implements a stochastic behaviour to induce variability in the collected data; choosing a different action with probability p and requesting the values of the slots in a different order. The corpus is collected using two different policy parameter sets.

4.2 General LSTM-based state tracker

A general dialogue state tracker, based on the model described on section 2.2, has been implemented. Each value filter is composed by a linear feedforward layer of size 20 and a LSTM (Hochreiter and Schmidhuber, 1997) layer of size 30. *Dropout* (Srivastava et al., 2014) regularisation is used in order to reduce overfitting with dropout rate of 0.2 in the input connections and 0.5 in the remaining non-recurrent connections. The models are trained for 60 iterations with stochastic gradient descent. A validation set consisting on 20% of the training data is used to choose the parameter set corresponding to the best iteration. Model combination is also used to avoid overfitting. Every model is trained with 3 different seeds, and 5 different parameter sets are saved for each seed, one for the best iteration in the first 20, and then another for the best iteration in each interval of 10 iterations.

4.2.1 ASR and phone related general features

In each turn t, each value-specific state tracker (filter) takes as input the value-specific input features $\mathbf{x}_v^t$. In this model, these correspond to the confidence score of the command related to the specific value, the confidence scores of the meta-commands such as "yes" or "no" and a one-hot encoding of the last system action. In addition, the models are evaluated concatenating the value specific features $\mathbf{x}_v^t$ with the following ASR and phone related general features $\mathbf{z}_v^t$:

•*ValAcc*: The ASR performance of the command corresponding to the value of the tracker can be used as general feature. In this paper, the accuracy per command is used, defining $\mathbf{z}_v^t$ as the estimated ASR accuracy of the value v.

•*PhSeq*: A weighted sequence of phones is generated form the ASR output (N-best list of commands) as described below. A pronunciation dictionary is used to translate each word into a sequence of one-hot encodings of phones (the size of the one-hot encoding is 45, as the number of phones in US English). Each of these encodings is weighted by the confidence score of that command in the N-best list. This sequence is fed into an RNN as explained in section 3.2.1, and $\mathbf{z}_v^t$ is de-

	Joint	Slot 1	Slot 2	Slot 3	Mean
Baseline	50.51%	81.00%	51.53%	55.72%	62.75%
General	68.87%	87.59%	66.57%	67.68%	73.95%
ValAcc	74.13%	88.90%	72.16%	66.59%	75.88%
PhSeq	68.38%	89.30%	66.20%	67.74%	74.41%
PostSeq	67.92%	89.20%	65.94%	67.61%	74.25%
ValPhEnc	57.93%	77.91%	61.56%	59.31%	66.26%
PhSeq-ValPhEnc	58.56%	79.85%	62.03%	58.97%	66.95%

Table 1: Joint, mean and per slot state tracking accuracy of trackers trained on *domain A* and tested on *domain B* for trackers using different features.

fined as the vector corresponding to the final state of this RNN. The RNN is composed by a single *GRU* (Chung et al., 2014) layer of size 15.

•*PostSeq*: A sequence of vectors (one vector per speech frame) with monophone-state level posterior probabilities are extracted from the output layers of a Deep Neural Network trained on the UASpeech corpus. The extracted vectors contain the posteriors of each of the 3 states (initial, central, and final) for the 45 phones of US English. To reduce the dimensionality of vectors, the posteriors of the each phone states are merged by summing them. To reduce the length of the sequence, the mean of each group of 10 speech frames is taken. This produces a sequence of vectors of size 45 and maximum length of 20, which is fed into an RNN in the same way as *PhSeq* features to obtain $\mathbf{z}_v^t$.

•*ValPhEnc*: For each value filter, $\mathbf{z}_v^t$ is defined as the 20 dimensional encoding of the sequence of phones of the command associated to the value v, extracted from the *seq2seq* model defined in section 3.2.2. The encoder and decoder RNNs of the seq2seq model are composed of two layers of 20 LSTM units and the model is trained on the *combilex* dictionary (Richmond et al., 2010).

Note that two different kinds of features can be distinguished; *value identity* features and *ASR output* features. Value identity features (*ValAcc* and *ValPhEnc*) give information about the value tracked. These features are different for each filter (as each filter has a different associated value), but they do not change over turns (time invariant). ASR output features (*PhSeq* and *PostSeq*), on the other hand, give information about the ASR output observed. They are the same for each filter but change in each dialogue turn.

5 Results

The results presented are the joint state tracking accuracy, the accuracy of each individual slot and the mean accuracy of the 3 slots. This is because it was found that the relation between the mean slot accuracy and the joint accuracy is highly non-linear, due to the high dependency on the ontology of the joint goals, while the costs optimized are related to the mean accuracy of the slots[7]. All the following numbers represent the average results for the models tested with the 6 simulated users described in sec. 4.1.1.

Table 1 presents the accuracy results for the model described in section 4.2, using only value specific general features (*General*) and using the different features described in section 4.2.1. The models are trained on data from *domain A* and evaluated on data from *domain B*. *Baseline* presents the state tracking accuracy for the rule-based state tracker presented in section 4.1.2. It can be seen that the *General* tracker outperforms the baseline by more than 10%, suggesting that the baseline tracker does not perform well in ASR challenging environments. As it is expected, including the accuracy estimates (*ValAcc*) outperforms all the other approaches, especially on the joint goal. Including *PhSeq* features has a slightly worse performance in the joint but outperforms the *General* features in the mean slot accuracy. Comparing the slot by slot results, it can be seen that *PhSeq* features outperform *General* features in slot 1 accuracy by almost 2% while having similar behaviour in the other 2 slots. *PostSeq* features have a performance very similar to *PhSeq*, suggesting that both features carry very similar information. Surprisingly, *ValPhEnc* and *PhSeq-*

[7]When joining the slot outputs, the "invalid goals" are discarded as described in section 4. Future work will explore how to join the slot outputs more efficiently.

	Joint	Slot 1	Slot 2	Slot 3	Mean
General	68.97%	87.81%	66.91%	67.55%	74.09%
PhSeq	69.27%	89.54%	66.14%	68.24%	74.64%
ValAcc	74.65%	89.62%	72.73%	67.24%	76.53%
ValPhEnc	72.98%	89.88%	72.87%	75.66%	79.47%
PhSeq-ValPhEnc	73.48%	91.61%	74.03%	77.20%	80.95%
ValId	60.83%	86.38%	66.95%	75.08%	76.14%

Table 2: Joint, mean and per-slot state tracking accuracy of trackers when including 200 dialogues from *domain B* in the training data.

ValPhEnc perform much worse than the other features. A detailed examination of the training results showed that, compared to *General* features, these features were performing about 10% better in the validation set (*domain A*) while getting 10% worse results in the test set (*domain B*). This suggests a strong case of overfitting to the training data, probably caused because the vocabulary size (36 words for train and other 36 words for test) is not large enough for the model to find similarities between the phone encoding vectors.

To partially deal with this problem, Table 2 shows the accuracy results when 200 dialogues from *domain B* are included in the training data. Including these dialogues in the training data has a very slight effect with the *General* and *PhSeq* features. *ValPhEnc* features, however, show a large improvement, outperforming *General* features by 4% in the joint goal and more than 5% in the mean slot accuracy. This improvement is seen in all the slots individually. To be sure that the model is not just learning the identities of the words, *ValId* features extend the *General* features including a one-hot encoding of the word identity. As it can be seen, even if the performance in the joint goal is very low the mean slot accuracy improves the performance of *General* features by 2%. However, it is still more than 3% below the *ValPhEnc* features, showing that *ValPhEnc* features are not just learning the value identity, they are effectively correlating the performance of values similar in the phone encoding space. Finally, including the concatenation of *PhSeq* and *ValPhEnc* features, outperforms all the other approaches, even *ValAcc* features for the mean slot accuracy by more than 4%.

6 Conclusions

This paper has shown how the generalisation to unseen states of a dialogue state tracker can be improved by extending the value specific features with ASR accuracy estimates. Using an RNN encoder jointly trained with the general state tracker to encode phone-related sequential features slightly improved state tracking generalisation. However, when the model was trained using dense representations of phone sequences encoded with a *seq2seq* model, the tracker strongly overfitted to the training data, even if *dropout* regularization and model combination was used. This might be caused by the small variability of the command vocabulary (36 commands in each domain), which was not large enough for the model to find useful correlations between phone encodings. When a small amount of data from the unseen domain was included into the training data, phone encodings greatly boosted performance. This showed that phone encodings are useful as dense representations of the phonetic structure of the command, helping the model correlate state tracking performance of values close in the phonetic encoding space. This method was tested on a single-word command-based environmental control interface, where slot-value accuracies can easily be estimated. In addition, in this domain, the sequences of phonetic features are usually short. However, this method could be adapted to larger spoken dialogue systems by estimating the concept error rate of the SLU output of concepts related to slot-value pairs. Longer phonetic feature sequences could also be used to detect "problematic phones", or correlate sentences with similar phonetic composition, given enough variability of the training dataset to avoid overfitting.

Acknowledgments

The research leading to these results was supported by the University of Sheffield studentship network PIPIN and EPSRC Programme Grant EP/I031022/1 (Natural Speech Technology).

References

I. Casanueva, H. Christensen, T. Hain, and P. Green. 2014. *Adaptive speech recognition and dialogue management for users with speech disorders*. Proceedings of Interspeech.

I. Casanueva, T. Hain, H. Christensen, R. Marxer, and P. Green 2015. *Knowledge transfer between speakers for personalised dialogue management.*. 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue

I. Casanueva, T. Hain, and P. Green 2016. *Improving generalisation to new speakers in spoken dialogue state tracking.*. Proceedings of Interspeech.

H. Christensen, S. Cunningham, C. Fox, P. Green, and T. Hain. 2012. *A comparative study of adaptive, automatic recognition of disordered speech*. Proceedings of Interspeech.

H. Christensen, I. Casanueva, S. Cunningham, P. Green, and T. Hain. 2013. *homeService: Voice-enabled assistive technology in the home using cloud-based automatic speech recognition*. Proceedings of SLPAT.

H. Christensen, I. Casanueva, S. Cunningham, P. Green, and T. Hain. 2014. *Automatic selection of speakers for improved acoustic modelling: recognition of disordered speech with sparse data*. Proceedings of SLT.

H. Christensen, M. B. Aniol, P. Bell, P. Green, T. Hain, S. King, and P. Swietojanski 2013. *Combining in-domain and out-of-domain speech data for automatic recognition of disordered speech*. proceedings of Interspeech.

H. Christensen, Nicolao, M., Cunningham, S., Deena, S., Green, P., and Hain, T. 2015. *Speech-Enabled Environmental Control in an AAL setting for people with Speech Disorders: a Case Study*. arXiv preprint arXiv:1604.04562.

K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio 2014. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. EMNLP.

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio 2014. *Empirical evaluation of gated recurrent neural networks on sequence modeling*. arXiv preprint arXiv:1412.3555.

M. Gašić and S. Young. 2014. *Gaussian Processes for POMDP-based dialogue manager opimisation*. IEEE Transactions on Audio, Speech and Language Processing.

M. Geist and O. Pietquin. 2011. *Managing uncertainty within the KTD framework*. Proceedings of JMLR.

K. Georgila, J. Henderson and O. Lemon. 2006. *User simulation for spoken dialogue systems: learning and evaluation*. proceedings of INTERSPEECH

M. Henderson, B. Thomson, and J. Williams. 2014. *The second dialog state tracking challenge*. 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue

M. Henderson, B. Thomson, and J. Williams. 2014. *The third dialog state tracking challenge*. Spoken Language Technology Workshop (SLT)

M. Henderson, B. Thomson and S. Young. 2014. *Word-Based Dialog State Tracking with Recurrent Neural Networks*. Proceedings of the SIGDIAL 2014 Conference

M. Henderson, B. Thomson and S. Young 2014. *Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation*. Spoken Language Technology Workshop (SLT)

S. Hochreiter and J. Schmidhuber. 1997. *Long short-term memory.*. Neural computation

H. Kim, M. Hasegawa-Johnson, A. Perlman, J. Gunderson, T. Huang, K. Watkin, and S. Frame. 2008. *Dysarthric speech database for universal access research*. Proceedings of Interspeech.

S. Lee and M. Eskenazi. 2013. *Recipe For Building Robust Spoken Dialog State Trackers: Dialog State Tracking Challenge System Description*. Proceedings of the SIGDIAL 2013 Conference

S. Lee 2013. *Structured discriminative model for dialog state tracking*. Proceedings of the SIGDIAL 2013 Conference

R. Lowe, Pow, N., Serban, I., and Pineau, J. 2015. *The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems*. SIGDial conference

N. Mrksic, D. O. Saghdha, B. Thomson, M. Gai, P. H. Su, D. Vandyke, and S. Young 2015. *Multi-domain dialog state tracking using recurrent neural networks*. arXiv preprint arXiv:1506.07190.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean 2013. *Distributed Representations of Words and Phrases and their Compositionality*. Advances in neural information processing systems.

K. Sun, L. Chen, S. Zhu and K. Yu. 2014. *The SJTU System for Dialog State Tracking Challenge 2*. Proceedings of the SIGDIAL 2014 Conference.

K. Sun, L. Chen, S. Zhu and K. Yu. 2014. *A generalized rule based tracker for dialogue state tracking*. Spoken Language Technology Workshop (SLT).

K. Richmond, R. Clark, and S. Fitt. 2010. *On generating combilex pronunciations via morphological analysis*. Proceedings of Interspeech.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov. 2014. *Dropout: A simple way to prevent neural networks from overfitting*. The Journal of Machine Learning Research

J. Schatzmann, B. Thomson, K. Weilhammer, H. Ye and S. Young. 2006. *Agenda-based user simulation for bootstrapping a POMDP dialogue system*. Human Language Technologies

B. Thomson, and S. Young. 2010. *Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems*. Computer Speech and Language.

B. Thomson, M. Gasic, M. Henderson, P. Tsiakoulis and S. Young. 2012. *N-best error simulation for training spoken dialogue systems*. Spoken Language Technology Workshop (SLT)

P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol 2008. *Extracting and composing robust features with denoising autoencoders*. Proceedings of the 25th international conference on Machine learning

Z. Wang, and O. Lemon. 2013. *A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information*. Proceedings of the SIGDIAL 2013 Conference

J. Williams and S. Young. 2007. *Partially observable Markov decision processes for spoken dialog systems*. Computer Speech and Language.

J. Williams. 2014. *Web-style Ranking and SLU Combination for Dialog State Tracking*. Proceedings of SIGDIAL.

J. Williams, A. Raux, D. Ramachandran, and A. Black. 2013. *The dialog state tracking challenge*. Proceedings of the SIGDIAL 2013 Conference

T. Wen, Gasic, M., Mrksic, N., Rojas-Barahona, L. M., Su, P. H., Ultes, S., and Young, S. 2016. *A Network-based End-to-End Trainable Task-oriented Dialogue System*. arXiv preprint arXiv:1604.04562.

S. Young, M. Gašić, B. Thomson and J. D. Williams. 2013. *POMDP-Based Statistical Spoken Dialog Systems: A Review*. Proceedings of the IEEE.