

# A DATA EFFICIENT END-TO-END SPOKEN LANGUAGE UNDERSTANDING ARCHITECTURE

Marco Dinarelli\*    Nikita Kapoor\*    Bassam Jabaian<sup>†</sup>    Laurent Besacier\*

\* LIG - Université Grenoble Alpes, France    <sup>†</sup>LIA- Avignon Université, France

## ABSTRACT

End-to-end architectures have been recently proposed for spoken language understanding (SLU) and semantic parsing. Based on a large amount of data, those models learn jointly acoustic and linguistic-sequential features. Such architectures give very good results in the context of domain, intent and slot detection, their application in a more complex semantic chunking and tagging task is less easy. For that, in many cases, models are combined with an external a language model to enhance their performance.

In this paper we introduce a data efficient system which is trained end-to-end, with no additional, pre-trained external module. One key feature of our approach is an incremental training procedure where acoustic, language and semantic models are trained sequentially one after the other. The proposed model has a reasonable size and achieves competitive results with respect to state-of-the-art while using a small training dataset. In particular, we reach 24.02% Concept Error Rate (CER) on MEDIA/test while training on MEDIA/train without any additional data.

**Index Terms**— End-to-End SLU, sequence-to-sequence models, joint learning, data efficiency, MEDIA corpus

## 1. INTRODUCTION

Spoken Language Understanding (SLU) aims at extracting a semantic representation from a speech signal in human-computer interaction applications [1]. First SLU systems were based on *pipeline* architectures where an automatic speech recognition (ASR) module generates a transcription of utterances and a SLU module predicts the semantic labels. *Pipeline* systems now tend to be replaced by *end-to-end*<sup>1</sup> architectures based on neural models, where semantic representations are produced directly from a speech input without using transcriptions [2, 3, 4, 5]. Most of recently proposed end-to-end models are based on sequence-to-sequence architectures. They were initially applied to speech translation [6, 7] and then to SLU tasks where the main goal is to extract the domain and user intent from an utterance, together with some semantic slots [2, 5].

In this paper we address end-to-end semantic chunking and tagging of spoken utterances. The most relevant works of the literature with respect to this task [3, 4] propose models based

on Feed-Forward Neural Networks (FFNN) similar to the Deep Speech 2 model proposed for ASR [8], and an independent pre-trained language model re-scores semantic outputs. Except for [5], most end-to-end SLU systems of the literature are trained on huge amount of data. [3] also apply pre-training and transfer learning from other NLP tasks such as Named Entity Recognition (NER).

The contribution of this paper lies in the proposal of a data efficient architecture which is trained end-to-end, with no additional pre-trained external module. The proposed model achieves competitive results with respect to state-of-the-art while using a small training dataset (French MEDIA [9]) and having a reasonable computational footprint. In particular, we reach 24.02% Concept Error Rate (CER) on MEDIA/test while training on MEDIA/train without any additional data.

The remainder of this paper is organised as follows. After presenting the task addressed by this work in Section 2, we describe our sequence-to-sequence neural model in Section 3. Section 4 provides our experimental study on the French MEDIA corpus and we conclude in Section 5.

## 2. SLU TASK ADDRESSED (MEDIA)

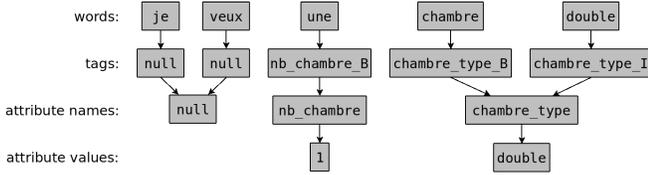
In this work we are interested in the task of semantic chunking and tagging of speech signals, corresponding to the user utterances in a conversation with a spoken dialog system. We focus on the specific domain of hotel information and reservation via an automatic system. This particular context is offered by the French MEDIA corpus [9]. It is made of 1 250 human-machine dialogs acquired with a *Wizard-of-OZ* approach, where 250 users followed 5 different reservation scenarios. Spoken data was manually transcribed and annotated with domain concepts, following a rich ontology. Statistics on the training, development and test data of the MEDIA corpus are shown in Table 1. We note that, while all turns have been manually transcribed (*total duration* in the table) and can be used to train ASR models, only user turns have been annotated with concepts (*user sentences*) and can be used to train SLU models.

Before the diffusion of neural networks, SLU was performed with pipeline systems [10, 11]. ASR was trained on large amount of data and refined on a specific SLU task. ASR transcripts were used as input to the SLU module, whose purpose was to tag words with the concepts. A further module was in charge to extract normalized values from tokens instantiating a particular concept. This is the processing applied also by a recent pipeline system

<sup>1</sup>Our approach, like previous approaches in the literature, is end-to-end at inference time, that is we do not use any intermediate representation between speech and semantic level at decoding; however we do use transcriptions at training time.

	Train		Dev		Test	
total duration	41.5 hours		3.5 hours		11.3 hours	
# user sentences	12,908		1,259		3,005	
	Words	Tags	Words	Tags	Words	Tags
# tokens	94,466	43,078	10,849	4,705	25,606	11,383
# types	2,210	99	838	66	1,276	78
OOV%	-	-	1.33	0.02	1.39	0.04

**Table 1.** Statistics of the MEDIA corpus



**Fig. 1.** Schema of concept (attribute names) and value (attribute values) extraction in the SLU task for spoken dialog systems [11]

based on neural networks [12]. The concept (attribute names) and value (attribute values) extraction schema is shown in Figure 1. In this work we focus on attribute names extraction only, and we decode directly whole concepts, without passing through the BIO intermediate format.

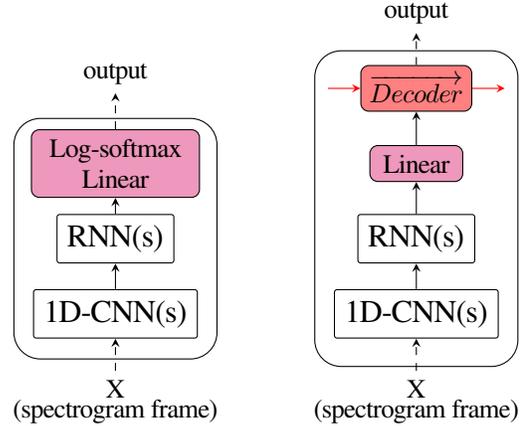
All these modules can be seen as different sub-tasks. Moreover the ASR can be further split into acoustic and language model learning sub-tasks. Thanks to neural networks, the sub-tasks can be learned jointly in an end-to-end framework. The literature also shows that it might be convenient to learn incrementally the different sub-tasks involved in SLU: acoustic features, characters, tokens and finally concepts. We adopt a similar *incremental* strategy learning different features at different learning stages. We use sequence-to-sequence neural models for learning jointly acoustic and linguistic features, without using any externally pre-trained language model to rescore local predictions from the acoustic feature encoder. Joint training is performed at different sub-task levels, eventually resulting in learning semantic features jointly with acoustic and linguistic features. Our neural models are detailed in the following section.

### 3. A DATA EFFICIENT SEQUENCE-TO-SEQUENCE MODEL FOR SLU

#### 3.1. Basic, Sequential and 2-Stage Models

The architecture proposed in this paper is based on sequence-to-sequence neural models [13, 14]. The encoder has a similar architecture as the one used in the *Deep Speech 2* architecture [8]. It takes as input the spectrogram of the speech signal, which is passed through a stack of convolutional and recurrent layers, and generates representations of the output. This can be characters, tokens or semantic classes, depending on which sub-task is targeted.

The decoder has the same architecture as in [15]. It has characteristics of both recurrent and *Transformer* [14] neural networks. It takes as input the output of the encoder, but also its own previous predictions. These are integrated into the decoder



**Fig. 2.** Basic Model (left), Sequential Model (right)

as embeddings of discrete items (indexes), the hidden layer of the decoder embeds thus a concatenation of both acoustic and linguistic/semantic features. Thanks to this choice, the architecture learns joint characteristics of both acoustic and language models, when characters or tokens are the items to be predicted. The model learns jointly acoustic and semantic features, when semantic tags are the items to be predicted. The fact that the current prediction depends also on previous predictions, allows the hidden layer to encode the sequential nature of output items.

**Basic (acoustic) model.** Since in general it is easier to learn acoustic and linguistic-sequential features incrementally, we use the encoder of our architecture as a *basic* model. In order to be trained individually, we add on top of it a linear layer with a *log-softmax* output function. A schema of this basic model is given in Figure 2 (left).

**Sequential (acoustic+language) model.** In order to obtain our sequence-to-sequence model, we replace the log-softmax with a decoder. A schema of this sequential model is given in the Figure 2 (right). We note that the basic model predicts an output item for each spectrogram frame independently, as a sequence of local decisions. The sequential model in contrast takes previous predicted items into account for the current prediction, and thus makes contextual decisions.

**2-stage (acoustic+language) model.** We use the same idea as [5] who proposed a model learning phones and tokens together. Our only difference is that instead of predicting phones, we decode characters. This has the advantage of being pronunciation dictionary free. Our 2-stage model is obtained by stacking a sequential model (cf. Figure 2, on the right) on top of a basic model (cf. figure 2, on the left).

**2-stage (acoustic+language+semantic) model.** The final SLU model is obtained by adding another decoder on top of a 2-stage model. It is trained to decode semantic concepts. A schema of our final SLU model is depicted in Figure 3.

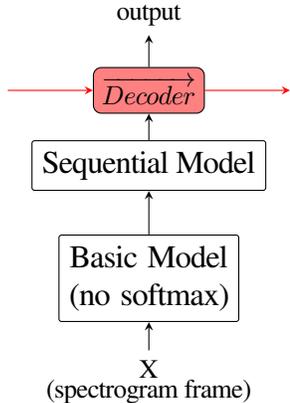


Fig. 3. Schema of our End-to-End SLU model

### 3.2. Incremental Training Strategy

We use the neural architectures introduced in the previous section for training incrementally, one after the other, all the sub-tasks involved in SLU: acoustic features, characters, tokens and concepts decoding from speech. We learn first a basic model for decoding characters, this is used as starting point for learning a sequential model for characters. The sequential model for characters is used as starting point for a basic 2-stage model decoding tokens, which in turn initializes parameters of a sequential 2-stage model. The latter decodes tokens, it learns jointly acoustic and linguistic features of tokens, together with token sequences. It thus performs at the same time the role of acoustic and language models of traditional ASR systems. Finally, a sequential model for decoding concepts (SLU) is learned by stacking a new decoder on top of a 2-stage model. All models are learned minimizing the *CTC* loss [16].

When learning our sequential model with gold items, previous items given as input to the decoder are a much stronger predictor of the current item compared to representations of spectrogram input. The model gives thus much more importance to the previous items than to acoustic features, creating a mismatch between training and testing conditions, when previous items must be predicted. In order to avoid this behavior, we use a similar strategy as [17]: sequential models are trained starting with predicted items, when the learning rate is the greatest. After a given number of training epochs (an hyper-parameter), when weights have sufficiently been shaped from acoustic features, we switch to training with gold items. The rest of the training is lead by the error rate on development data (see section 4.1).

Another learning problem may be introduced by the very different length between input sequence (speech spectrograms) and gold output sequences (characters, tokens or concepts). Let the input sequence have length  $N$  and the output sequence have length  $M$ . In general  $N \gg M$ .<sup>2</sup> When the decoder is at processing step  $i$ , it has no information on which spectrogram frames to use as input. This problem can be solved using an attention mechanism [18] to

<sup>2</sup>In our data we found that  $\frac{N}{M} \leq 30$

Model	Dev WER	Test WER
Basic Char	27.74 (*)	–
Seq Char	24.05 (*)	23.32 (*)
Basic Tok	30.79	–
Seq Tok	29.42	28.71
Basic 2-Stage	28.00	–
Seq 2-Stage	<b>27.95</b>	<b>27.01</b>
Seq 2-Stage (no-incremental)	63.70	63.61
Seq 2-Stage (no-curriculum)	28.15	27.96

Table 2. ASR Results on MEDIA - (\*) is a character error rate

focus on the correct part of the input sequence depending on the part of the output sequence being decoded. However, in this work we propose a simpler but efficient solution (inspired by [19, 20, 21]), based on a basic mechanism (since alignment is monotonic in SLU): we compute the ratio between output and input sequence lengths  $r = \frac{M}{N}$ , and when the model decodes at position  $i$ , it uses the sum of the encoder states around position  $\lfloor i \cdot r \rfloor$ .

We further improve our training procedure with a variant of the curriculum strategy used in [8]. We sort speech turns based on their increasing length. Shorter turns, which have simpler sequential structures, are presented first to the model. After a given number of training epochs (an hyper-parameter), we switch to training with whole-dialog turn sequences.

## 4. EVALUATION

### 4.1. Settings

The size of layers in our models, resulted from optimization on the Dev data, are as follows: the input spectrogram features are of dimension 81 and so is the dimension of convolutional layers, recurrent layers (LSTMs) have 256 dimensions. In the decoder, embeddings of previous predictions have 150 dimensions while hidden layers have 300. The decoder predicting concepts has twice more dimensions for each layer. Our basic and sequential models use only 1 CNN layer with stride 2 and 2 Bi-LSTM layers, in contrast to [4, 3] where 2 and 6 are used, respectively. *Layer normalization* and *Dropout* regularization [22] (with  $p=0.5$ ) are applied between each two layers. Our most complex model (see section 4.2) has less than 9.8M parameters, in contrast to e.g. [23] with 97M. All models are learned with an *ADAM* optimizer [24], with learning rate of 0.0005 decayed linearly over 60 epochs. The training procedure starts with the incremental training strategy described in previous section and using predicted items. After 5 epochs we switch to gold items. At this point, each time the error rate is not improved on DEV data for 2 consecutive epochs, we switch between gold and predicted items learning.

### 4.2. Results

We evaluate both ASR (Word Error Rate) and SLU (Concept Error Rate) results on MEDIA corpus (Dev and Test).

Model	Training Speech	Dev CER	Test CER
Seq 2-Stage + $\overrightarrow{Dec}$	41.5h	28.11	27.52
Seq 2-Stage + $\overrightarrow{Dec}$ tune	41.5h	28.18	27.35
Seq 2-Stage + $\overrightarrow{Dec}$ XT	41.5h	<b>23.39</b>	<b>24.02</b>
State-of-the-art Models			
E2E SLU [4]	300h	30.1	27.0
E2E Baseline [3]	41.5h	–	39.8
E2E SLU [3]	500h	–	23.7
E2E SLU + curr. [3]	500h	–	16.4

**Table 3.** SLU Results on MEDIA. For full comparison we report the best result of [3] (16.4), which is obtained with a beam-search decoding, while the others are obtained, like our results, with greedy decoding.

**ASR** results are presented in Table 2. Together with the character and the 2-stage models, we show performance from a model decoding directly tokens. We observe that the sequential model outperforms the basic model, which does not use information of the output’s sequential structure. The 2-stage model always outperforms the token model which demonstrates that using pre-trained character models gives an advantage over training directly for decoding tokens. Training incrementally the different stages of the model is the most effective choice: training a 2-stage model from scratch (*no-incremental* in the table), the error rate is much higher (over 60%). Finally, using the curriculum learning (sort speech turns based on their increasing length) proves also to be slightly beneficial: 1% lower WER compared to a model trained without curriculum strategy (*no-curriculum* in the table). Our best results are competitive with previously published ASR performance on MEDIA: ASR used in [11] had an error rate of 30.4 on Dev data, which we improve by a large margin. Our own ASR baseline based on an HMM-DNN model trained with *Kaldi*<sup>3</sup> reached an error rate of 25.1 on Dev. The best results shown in table 2 are not too far, and they provide the advantage of being trained end-to-end without any external data nor language model. Better ASR performances were published lately on MEDIA [12, 3] but these were trained on up to 12 times more ASR training data. In particular [3] trains the ASR part of the model with 4 different corpora, for a total of roughly 300 hours of speech. [3] used 5 different corpora, accounting for 500 hours of speech. [12] uses even more data. Our system is trained on MEDIA training data only, consisting of 41.5 hours of speech. A comparison of the amount of speech training data used for end-to-end ASR systems on MEDIA is given in the column *Training Speech* of table 3.<sup>4</sup>

**SLU** performances are given in Table 3. Our results can be compared with some previous works [4, 3]. We note however that results reported in [4, 3] are obtained with models trained with

<sup>3</sup><https://kaldi-asr.org/>

<sup>4</sup>The amount of speech data for training the SLU is not always detailed in those papers, it is generally smaller or equal to the amount for training ASR reported in the table.

much more data exploiting NER tasks with transfer learning. In particular [3] uses 3 NER corpora for bootstrapping an end-to-end system. This is then fine-tuned on a first SLU corpus similar to MEDIA, and finally on MEDIA (see [3] for details). For training our system, we note that only user turns are annotated with concepts, these account for 16.8 hours, that is less than half of the 41.5 hours of speech available, containing both machine and user turns. The only result that is obtained in similar training conditions as ours, is the baseline model of [3]. We can see that our model improves such baseline by a large margin, proving that learning jointly acoustic and linguistic-sequential features in an end-to-end framework is more effective than rescoring outputs with an independent language model. More importantly, our results are comparable to the best results reported in [4]. Once again these are obtained with models trained with much more data and a curriculum strategy via transfer learning among different corpora. This outcome highlight even more our data efficient training procedure.

Our SLU results in Table 3 are obtained with a model decoding both concepts and tokens. This choice is imposed by the need of keeping track of which words instantiate a concept. Using the example in Figure 1, the words “*chambre double*” (double room) for instance, instantiate the concept **chambre-type**. Our model, similar to [4], generates the output  $\langle \textit{chambre double} \textit{chambre-type} \rangle$ , which allows for attribute value extraction.<sup>5</sup> This choice constrains the model to learn chunking and tagging at the same time, which is a much harder problem than just tagging [1]. To improve this, we propose two alternatives: (1) refining token representations during SLU model training (*tune* in the table) and (2) decoupling chunking and tagging using a second decoder which decodes only concepts (*XT* in the table for *extended*). In this latter decoding strategy, the first decoder generates a first output with concept boundary annotation while the second decoder generates concepts only, aligned to the output of the previous decoder. As we can see in the results, the *XT* model obtains results comparable to those obtained by models trained with much more training data.

## 5. CONCLUSIONS

In this paper we proposed a data efficient end-to-end SLU system. Our model learns jointly acoustic and linguistic-sequential features, allowing to train SLU models without an explicit language model module (pre-trained independently and/or on huge amount of data). The efficiency of our system comes mostly from an incremental training procedure. The proposed model achieves competitive results with respect to state-of-the-art while using a small training dataset and having a reasonable computational footprint.

## 6. ACKNOWLEDGEMENTS

This work was funded by the Institute Carnot Cognition.

<sup>5</sup>In this work we focus however on concept extraction only

## 7. REFERENCES

- [1] Renato De Mori, *Spoken Dialogues with Computers*, Academic Press, Inc., Orlando, FL, USA, 1997.
- [2] D. Serdyuk, Y. Wang, C. Fuegen, A. Kumar, B. Liu, and Y. Bengio, “Towards end-to-end spoken language understanding,” *CoRR*, vol. abs/1802.08395, 2018.
- [3] A. Caubrière, N. A. Tomashenko, A. Laurent, E. Morin, N. Camelin, and Y. Estève, “Curriculum-based transfer learning for an effective end-to-end spoken language understanding and domain portability,” *CoRR*, vol. abs/1906.07601, 2019.
- [4] S. Ghannay, A. Caubrière, Y. Estève, N. Camelin, E. Simonnet, A. Laurent, and E. Morin, “End-to-end named entity and semantic concept extraction from speech,” in *IEEE SLT Workshop*, Athens, Greece, Dec. 2018.
- [5] L. Lugosch, M. Ravanelli, P. Ignoto, V. S. Tomar, and Y. Bengio, “Speech model pre-training for end-to-end spoken language understanding,” *CoRR*, vol. abs/1904.03670, 2019.
- [6] A. Berard, O. Pietquin, C. Servan, and L. Besacier, “Listen and translate: A proof of concept for end-to-end speech-to-text translation,” in *NIPS Workshop on End-to-end Learning for Speech and Audio Processing*, 2016.
- [7] R. J. Weiss, J. Chorowski, N. Jaitly, Y. Wu, and Z. Chen, “Sequence-to-sequence models can directly transcribe foreign speech,” *arXiv preprint arXiv:1703.08581*, 2017.
- [8] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos, E. Elsen, J. H. Engel, L. Fan, C. Fougner, T. Han, A. Y. Hannun, B. Jun, P. LeGresley, L. Lin, S. Narang, A. Y. Ng, S. Ozair, R. Prenger, J. Raiman, S. Satheesh, D. Seetapun, S. Sengupta, Y. Wang, Z. Wang, C. Wang, B. Xiao, D. Yogatama, J. Zhan, and Z. Zhu, “Deep speech 2: End-to-end speech recognition in english and mandarin,” *CoRR*, vol. abs/1512.02595, 2015.
- [9] H. Bonneau-Maynard, S. Rosset, C. Ayache, A. Kuhn, and D. Mostefa, “Semantic annotation of the french media dialog corpus,” in *ISCA Eurospeech*, Lisboa, Portugal, 2005.
- [10] M. Dinarelli, A. Moschitti, and G. Riccardi, “Concept segmentation and labeling for conversational speech,” in *Proceedings of Interspeech*, 2009.
- [11] S. Hahn, M. Dinarelli, C. Raymond, F. Lefèvre, P. Lehen, R. De Mori, A. Moschitti, H. Ney, and G. Riccardi, “Comparing stochastic approaches to spoken language understanding in multiple languages,” *IEEE TASLP*, vol. 99, 2010.
- [12] E. Simonnet, S. Ghannay, N. Camelin, Y. Estève, and R. De Mori, “ASR error management for improving spoken language understanding,” *CoRR*, vol. abs/1705.09515, 2017.
- [13] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Proceedings of NIPS*, 2014.
- [14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2017.
- [15] M. Dinarelli and L. Grobol, “Hybrid neural models for sequence modelling: The best of three worlds,” *CoRR*, 2019.
- [16] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in *Proceedings of ICML*, 2006, pp. 369–376, ACM.
- [17] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, “Scheduled sampling for sequence prediction with recurrent neural networks,” *CoRR*, vol. abs/1506.03099, 2015.
- [18] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *CoRR*, vol. abs/1409.0473, 2014.
- [19] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, Nov. 1997.
- [20] A. Joulin and T. Mikolov, “Inferring algorithmic patterns with stack-augmented recurrent nets,” *CoRR*, vol. abs/1503.01007, 2015.
- [21] G. Weiss, Y. Goldberg, and E. Yahav, “On the practical computational power of finite precision rnns for language recognition,” *CoRR*, vol. abs/1805.04908, 2018.
- [22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *Journal of Machine Learning Research*, vol. 15, 2014.
- [23] P. Haghani, A. Narayanan, M. Bacchiani, G. Chuang, N. Gaur, P. Moreno, R. Prabhavalkar, Z. Qu, and A. Waters, “From audio to semantics: Approaches to end-to-end spoken language understanding,” *CoRR*, vol. abs/1809.09190, 2018.
- [24] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *International Conference for Learning Representations*, 2015, Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.