Improving Dialogue Act Classification by Considering Context: An investigation of dialogues acts using Bert Encoding (Devlin et al.)

Mathieu Garrouty*
ENSAE
Repository of the project
mathieu.garrouty@ensae.fr

Louise Carrotte*
ENSAE
louise.carrotte@ensae.fr

Abstract

Dialog Act Recognition became a major field of study within the last years, especially due to the growth of chat assistant such as ChatGPT. In order to generate a conversation as "human-friendly" as possible, the vital point is to understand the purpose attached to messages. In this paper, we focus on classifying a dataset made of multi-turn dialogs of two people, labelled with their Dialog Act. Our goal is to predict the Dialog Act Classification on our training test. We compare a basic sequence-level model (Neural Network learning from all sequences labelled) to dialog-level ones, allowing to take into account the context of a sequence. We use Recurrent Neural Network, both with and without self-attention mechanisms. Within a comparable training time, we have a significant increase in the accuracy of our prediction, showing the importance of the context for a better representation of dialogs in NLP.

1 Problem Framing

Our aim is to build a Sequence Classifier for dialogs acts. In linguistics and in particular in natural language understanding, a **dialog act** is an utterance, in the context of a conversational dialog, that serves a function in the dialog. Types of dialog acts include a question, a statement, or a request for action (McTear et al., 2016). Each dialog involves two speakers, speaking turn by turn.

Dataset

To do so, we will use the **DailyDialog** Corpus (Li, 2017), which is a human-written multiturn dialogue dataset, reflecting our daily communication way and covering various topics about our daily life. The dataset is already splitted into train, validation and test set.

Total Dialogues	13,118
Average Speaker Turns Per Dialogue	7.9
Average Tokens per Dialogue	114.7
Average Tokens Per Utterance	14.6

Table 1: Statistics of DailyDialog Dataset

Dialogs in our database are multi-turned, and always involve only two speakers. Each speaker can pronounce several sentences in a single sequence. We can represent a Dialog D as follow

$$D = (S_1^a, S_1^b, S_2^a, S_2^b, S_3^a, ..)$$

, where S_i^j represents a sequence. A sequence is represented as follow

$$S_i^j = (s_1^{i,j}, s_2^{i,j}, ..)$$

, where $s_k^{i,j}$ represents a sentence. Eventually, a sentence is made of utterances :

$$s_k^{i,j} = (u_1^{k,i,j}, u_2^{k,i,j}, u_3^{k,i,j}, ..)$$

Each sequence is manually labelled with its nature (inform (1), question (2), directive (3), commissive (4)), 0 being used as a dummy variable.

Dialog Act	Count	Percentage
Informative	39873	45.7
Question	24974	28.6
Directive	14242	16.3
Commissive	8081	9.4

Table 2: Dialog Act Repartition in Dailydialog

Sequence	Act
”Say, Jim, how about going for a few beers after dinner?”	3
”You know that is tempting but is really not good for our fitness.”	4
”What do you mean? It will help us to relax.”	2

Table 3: Extract of a dialog

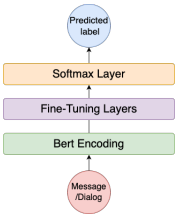

Figure 1: Generic Network Architecture

2 Experiments Protocol

To build our Dialog Act Classifier, we use Neural Network Architecture, widely used in Natural Language Recognition. Especially, we are working with *PyTorch* library on Python. In this paper, we use the notation (X_i, Y_i) for data, where X_i can either represent a dialog (list of messages) or a message, and Y_i either a list of labels (Dialog Acts) or a label.

Our models are trained with these data, and are ranked according to their accuracy on the test set :

$$Acc = \frac{1}{|TestSet|} \sum_{X_i in TestSet} \mathbb{1}_{\hat{Y}_i = Y_i}$$

Where $\hat{Y}_i$ is the label predicted by our model. In order to compare message and dialog level accuracy, we always compute the accuracy at a message level. We are using accuracy to have results that are not relying on the loss used for training. As a consequence, we use both NLLLoss and CrossEntropyLoss in our models, depending on which provides the best accuracy.

Data Encoding

The first step of each NLP task consists in transforming language into vectors. They are several ways to do so, and we consider the **BERT** model (Horev, 2018). It is an open-source NLP pretrained model powered by Google, transforming each word into a vector. The original model turns each token into a vector in $\mathbb{R}^{768}$. For computational efficience, we consider the *Bert tiny* library, allowing a dimension-reduced representation in $\mathbb{R}^{128}$.

Final Layer of our Network

As we are working on classification task, all our models have the same final layer, consisting of a Softmax Layer, to compute a probability score for each label. We then take the highest probability label.

Loss Function

We choose the CrossEntropy Loss function, givent that it gives the best accuracy result. This function is widely used in classification problem. For an input X, with a softmax vector $\hat{Y} = (p_1, p_2, p_3, p_4)$, where (p_i) represents the computed probability of i being the right label for X, the value of the loss function is :

$$L(X, \hat{Y}) = \sum_{i=1}^{4} -y_i log(p_i)$$

where $Y = (y_i)$ is the vector of the real label of X ($y_i = 1$ only if label of X is i).

Dropout rate

In order to prevent our model from overfitting, we add a dropout layer (consisting in ignoring a share p of our data). This share, called the *dropout rate*, is chosen by trying several numbers and selecting the one giving the best accuracy **on the validation dataset**. Indeed, we can have a signficant difference between the accuracy on the training and the validation set.

2.1 Message-level classification

For our baseline, we flatten our dataset of dialog into a dataset of messages. We encode them using *Bert tiny*, and we add two linear layers before classification. Formally, for a message $S = (s_1, s_2, s_3, ...)$, where $s_i = (u_1^i, u_2^i, ...)$ denotes a sentence, and u denotes a token, our bert encoder returns a vector from $\mathbb{R}^{128}$ for each token. To obtain a vector for our sentence, we choose to average the vectors associated to each word of the sentence.

2.2 Dialog level classification

We now consider a dialog-level, inducing a hierarchical architecture (Colombo et al., 2020) . We now feed our models with matrices of $\mathbb{R}^{12*128}$, where each row represents a message. Each message is represented by the average of the embed-

dings of its words, and 12 is the max length of dialog considered. We add rows of zeros for shorter dialogs.

2.2.1 Linear Layer

After our encoding, we add two linear layers before classification.

2.2.2 BiLTSM Layer

After our encoding, we add a Bidirectionnal Recursive Layer, to let the model learn the links between the rows of the input matrice.

3 Results

We implement our different methods. For each one, we tried several hyperparameters (learning rate, number of epochs), in order to maximize our accuracy, within computation time constraint.

3.1 Baseline

For our baseline, we find the best accuracy for a learning rate of 10^{-4}, and the loss (best accuracy with CrossEntropy) is not significantly decreasing after 7-8 epochs (we ran over 10), as shown on the graph :

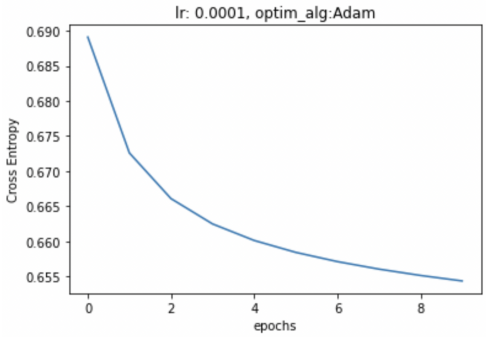

Figure 2: Loss on validation test for Baseline Model

	Precision	Recall	f1-score	Support
Inform	0.32	0.88	0.47	2948
Question	0.84	0.88	0.86	2175
Directive	0.59	0.48	0.53	1705
Commissive	0.49	0.08	0.14	867
Micro-averaged	0.45	0.70	0.55	7695
Macro-averaged	0.56	0.58	0.50	7695
Weighted-averaged	0.54	0.70	0.55	7695

Table 4: Classification report for Baseline Model

We end with an average accuracy of 0.452 with this model.

3.2 Linear Layer

We are now working at a dialog-level here, slightly modifying our data preprocessing. We have the same hyperparameters as for our baseline

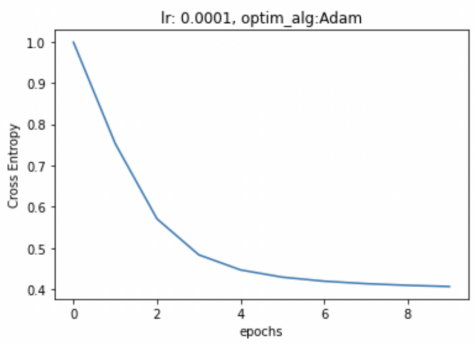

Figure 3: Loss on validation test for Linear Model

	Precision	Recall	f1-score	Support
Inform	0.66	0.88	0.76	2948
Question	0.82	0.91	0.86	2175
Directive	0.58	0.43	0.50	1705
Commissive	0.46	0.06	0.10	867
Micro-averaged	0.70	0.70	0.70	7695
Macro-averaged	0.63	0.57	0.55	7695
Weighted-averaged	0.67	0.70	0.66	7695

Table 5: Classification report for linear Model

The average accuracy on our validation test is 0.807.

3.3 BiLTSM Model

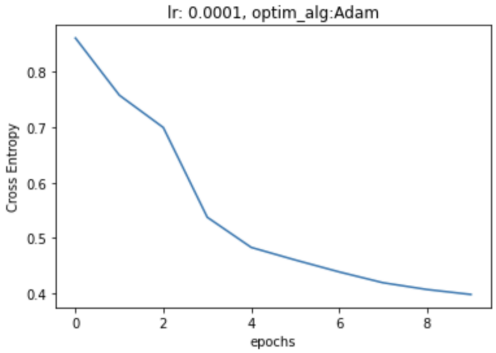

Figure 4: Loss on validation test for BiLTSM Model

For this model, we replace the former linear layers by a Bidirectionnal LTSM. These layers allow to backpropagate data in the Network, but they require more computational power. The average accuracy on our validation test is 0.804.

	Precision	Recall	f1-score	Support
Inform	0.67	0.89	0.76	2948
Question	0.81	0.92	0.86	2175
Directive	0.57	0.41	0.48	1705
Commissive	0.51	0.03	0.06	867
Micro-averaged	0.70	0.70	0.70	7695
Macro-averaged	0.64	0.56	0.54	7695
Weighted-averaged	0.67	0.70	0.65	7695

Table 6: Classification report for BiLTSM Model

4 Discussion/Conclusion

4.1 Overall results on accuracy

Despite the quality of Bert encoding, the accuracy of our baseline model was less than 0.5. We obtained a significant increase when we took our data to a dialog-level (30 percents). What can we learn from this result ?

First, it shows that Dialog Act Recognition is not only a message-level issue. Indeed, the increase of accuracy is due to the fact that we take into account the whole context of the message. Moreover, this supplement of information can be learnt from by a Neural Network.

However, we don't have a significant increase when using a BiLTSM model at a dialog-level, instead of linear layers. It can be due to the fact that messages are already vectorized at a message-level (we use the average of embedding of each words), so it is not possible to create links between messages and uterrances. Doing so would require bigger computational capacities.

Thanks to the classification report, we see that most of the difference of accuracy comes from the precision in classifying inform messages, which represents the biggest label.

Eventually, while finetuning our model, we have seen the importance of adding a **dropout layer**. Indeed, we have seen a difference in accuracy over 5 percent. It can show that our models might have too many parameters. For the linear models, it can be due to the two linear layers added for fine-tuning Bert.

4.2 Extension

To extend our results, it would be useful to try our models on different datasets. It might be interesting to adapt our last model (Bert + BiLTSM) to add self-attention mechanisms, by keep working at a word-level embedding. It would require greater computational capacities.

We could also try to use it to classify code-switched dialog, (Chapuis et al., 2021), requiring another layer for classifying language.

What could also be done would be to pre-train our encoders ourselves. (Chapuis et al., 2020)

Another possibility would be to use our modeld for different classification tasks. For instance, the database we used also provide the emotion attached to each message, classified according to the "bigSix Theory" (Ekman, 1992). However, data distribution is strongly unbalanced among the dataset :

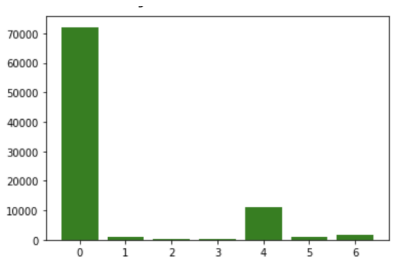

Figure 5: Distribution of emotion label in dailydialog

It would be possible to use it, but we should specify the weights of our Network to deal with unbalanced data.

A last extension would be to try to classify Dialog Act and Emotion in the same time, to check whether cross-information can be useful to increase the accuracy.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding.

Paul Ekman. 1992. An argument for basic emotions. *Cognition emotion*.

Michael McTear, Zoraida Callejas, and David Griol. 2016. The conversational interface: Talking to smart devices.

Yanran Li. 2017. A manually labelled multi-turn dialogue dataset.

Rani Horev. 2018. Bert model.

Pierre Colombo, Emile Chapuis, Matteo Manica, Emmanuel Vignon, Giovanna Varni, and Chloé Clavel. 2020. Guiding attention in sequence-to-sequence models for dialogue act prediction.

Emile Chapuis, Pierre Colombo, Matteo Manica, Matthieu Labeau, and Chloe Clavel. 2020. Hierarchical pre-training for sequence labelling in spoken dialog.

Emile Chapuis, Pierre Colombo, Matthieu Labeau, and Chloé Clavel. 2021. Code-switched inspired losses for generic spoken dialog representations.