Automatic Labeling of Problem-Solving Dialogues for Computational Microgenetic Learning Analytics

Yuanliang Meng*, Anna Rumshisky*, Florence Sullivan†, Kevin Keith†
*Text Machine Lab, Department of Computer Science, University of Massachusetts Lowell
†College of Education, University of Massachusetts Amherst
{ymeng, arum}@cs.uml.edu
fsullivan@educ.umass.edu, kevinkeith@landmark.edu

Abstract

This paper presents a recurrent neural network model to automate the analysis of students' computational thinking in problem-solving dialogue. We have collected and annotated dialogue transcripts from middle school students solving a robotics challenge, and each dialogue turn is assigned a code. We use sentence embeddings and speaker identities as features, and experiment with linear chain CRFs and RNNs with a CRF layer (LSTM-CRF). Both the linear chain CRF model and the LSTM-CRF model outperform the naïve baselines by a large margin, and LSTM-CRF has an edge between the two. To our knowledge, this is the first study on dialogue segment annotation using neural network models. This study is also a stepping-stone to automating the microgenetic analysis of cognitive interactions between students.

Keywords: dialogue act annotation, recurrent neural networks, microgenetic learning analytics

1. Introduction

Microgenetic analysis is an observational research technique in which the researcher attends closely to the discourse interactions and the use of tools within the learning environment in order to understand the genesis (or the origins) of cognitive change (Wertsch, 1991). It has a broad impact on the design and enactment of curriculum, the design of learning environments, as well as pedagogical practices. Microgenetic Learning Analytics (MLA) (Sullivan et al., 2015) requires the collection and analysis of all interactions among students over a given period of time. It has been noted that the robustness of microgenetic analysis derives from high-density observation (Siegler, 2006). However, it is the time-consuming nature of collecting and analyzing high-density observations that has restricted the application of microgenetic analysis to small case studies.

This paper is an initial attempt to identify computational methods that will allow for automation of MLA of conversational data. Such automation will transform educational researchers' ability to perform microgenetic analysis.

This paper presents a pilot study to automate the analysis of cognitive interactions in dialogue. We have transcripts of the dialogues from middle school students solving a robotics challenge (Sullivan et al., 2015). Dialogue turns (utterances) are annotated with carefully designed labels, known as the "computational thinking codes (CT codes)". We need to build computational models to assign the codes to dialogue turns automatically. In a broader sense, this is a dialogue action (DA) annotation task, with multiple agents in the dialogue. It is also related to the sequential sentence labeling task, in which a series of sentences need to be labeled.

2. Related Work

Many different classification algorithms have been used for dialogue act annotation. The most popular ones use support vector machines (SVM) (Margolis et al., 2010; Tavafi et al., 2013), Hidden Markov models (HMM) (Kim et al., 2010; Tavafi et al., 2013), conditional random fields (CRF) (Kim et al., 2010; Tavafi et al., 2013), and decision trees (Moldovan et al., ; Samei et al., 2014). These methods require handcrafted features, and their performance depends on the application domain. Few if any works to date have used the recently popular neural network-based approaches for dialogue segment labeling.

We can also consider DA annotation a special case of sequential sentence labeling, in which every sentence is an utterance. Existing systems for sequential sentence labeling are mostly based on traditional statistical machine learning methods too, such as Naïve Bayes (Huang et al., 2013), SVM (McKnight and Srinivasan, 2003; Hirohata et al., 2008), HMM (Lin et al., 2006), and CRF (Hirohata et al., 2008; Hassanzadeh et al., 2014). Recently, popular approaches use neural networks with word embeddings (Socher et al., 2013; Kim, 2014) and/or character embeddings (Zhang et al., 2015; Conneau et al., 2016) to train sentence encoders, and then perform classification. Those neural networks use either convolutional layers or recurrent layers to learn deep representations, and often produce better results than older systems. However, one drawback of most of the models is that they do not make use of context, but focus on representing each sentence independently (Dernoncourt et al., 2016). Moreover, the sequence of labels are not directly modeled like in CRF, although RNNs can capture that indirectly.

Dernoncourt et al. (2016) tries to combine the properties of RNN and CRF. They use character and token embeddings to train a sentence encoder, and use an output sequence optimization layer to incorporate transition probabilities of labels. However, their data is from medical paper abstracts, which have a strong tendency to follow certain styles of writing, and are much shorter than dialogues. They only have 5 classes, all with very distinctive semantic properties. Our 10 labels demand deeper understanding to be distinguished.

In fact, the same technique has been used in name entity recognition and other token-based sequence tagging tasks (Huang et al., 2015; Dernoncourt et al., 2017). The sequence optimization layer is also called a CRF layer. Sentences are much longer than name entities and contain more complex meanings, and thus sequential sentence labeling is a more difficult task.

3. Dataset

The dataset consists of dialogues of middle school students solving a robotics challenge together. There are two collaborative teams. Team A has 1 boy and 2 girls, and Team B has 2 boys. All students wore wireless microphones, and their interactions were videotaped. In this study, we obtained the transcribed dialogues and use text data for analysis.

In order to use RNNs to process sequences of sentences, we need to train a sentence encoder. Given the small dataset, we also tried external resources. Conneau et al. (2017) proposed a model to produce "universal sentence representations". They train sentence encoders on Natural Language Inference (NLI) task and claim such a process involves high-level understanding of text, and thus generates better sentence embeddings than other supervised or unsupervised learning methods. In our experiments, we use their pre-trained sentence encoder to represent sentences. The encoder is trained on SNLI (Bowman et al., 2015) and MultiNLI (Williams et al., 2017).

3.1. Computational Thinking Coding Scheme

The dialogue of each team is segmented into 20-minute conversation chunks. Eventually we have 10 chunks for Team A and 9 for Team B. Then the dialogue turns of the 19 files are fully annotated. We developed a computational thinking coding scheme, as shown in Table 1.

Code	Category	Description
A	Analysis	Planning, developing ideas
ATO	Algorithmic thinking -operation	Discussing programming functions
ATV	Algorithmic thinking -variable	Discussing quantities to use
D	Design	Building additions to the robot
DO	Debugging observations	Describing movement of robot
NSTO	Non-specific test outcome	
QB	Query building	Questions about using the Legos
QS	Query software	Ques. about software capabilities
QR	Query robot	Ques. about operations of robot
O	Other	Anything else

Table 1: Computational thinking coding scheme.

Every dialogue turn is labeled with one of the codes in Table 1. A dialogue turn is an utterance generated by a speaker. Occasionally a turn is split and assigned with two different labels, when the speaker expresses two ideas. In this case we describe it as two dialogue turns from the same speaker. In total we annotated 5723 dialogue turns from the 19 files. Utterances from individuals who are not the team members are excluded from analysis.

The code "QB" does not exist in any data we use, so it is ignored. The remaining classes are heavily imbalanced. For example, 58% of the dialogue turns have "O" label but only 0.6% have the "QS" label. Some techniques are commonly used to deal with imbalanced data, such downsampling, upsampling, setting weights etc., driven by specific purposes. In our experiment, we do not employ any of the techniques, but train our models on raw data.

3.2. 5-folds cross validation

Since the dataset is relatively small and the classes are imbalanced, we use 5-fold cross validation to evaluate the final results. In each iteration of training/testing, 15 out of 19 files are chosen as training set, and the remaining 4 files as test set. In the five iterations, the test sets are all different without overlapping, except that one file is used twice (because $4 \times 5 = 20$).

One may argue the files reflect different stages of the robotics challenge, and are not independent. However, we investigate the data in details and find local context much more relevant than long-distance context, so treating the files independently should not create significant errors.

Hyper-parameters are tuned on one split of data only. The 15 training files are divided as training and validation set by 13:2.

4. Experiments

We built two models. One is a conventional CRF model, and the other is an RNN with a CRF layer.

4.1. Linear Chain CRF model

Until very recently, linear chain CRFs have been the preferred choice for sequence labelling tasks in NLP. We ran this model as a baseline to compare to. This model estimates the conditional probability of a label sequence, assuming that a label at a certain time step depends on its preceding label as well as input data. The objective is to find the feature weights that maximize the conditional probability as shown in equation 1.

$$p(\mathbf{y}|\mathbf{x}) \propto \exp\left(\sum_{t=1}^{T}\left(\sum_{i=1}^{D} w_i f_i(y_t, y_{t-1}, X)\right)\right) \tag{1}$$

In our case, every time step t is corresponding to a dialogue turn with a label. The input X can be the whole series of input data although in practice we only use a portion of the whole input at each time step. Naturally, we can use the text of each dialogue turn, and possibly of nearby turns. Therefore we rewrite the feature function as $f_i(y_t, y_{t-1}, x_t)$.

Here f_i uses the previous label y_{t-1}, current label y_t and current input text representation x_t as input. D is the dimensionality of the feature space. w_i is the set of parameters to be trained. In order to make this model a baseline for direct comparison, we used a pre-trained model to generate sentence embeddings, folllowing the work of Conneau et al. (2017). Each sentence is represented as a 4096-dimensional vector. In addition to that, we also add another feature to indicate if the speaker has changed from the previous time step. As a result, the feature space has 4097 components. In our dataset, two consecutive turns are from different speakers in the vast majority of cases, but occasionally an utterance from the same speaker are split, usually due to a change of idea. Our implementation used the Python wrapper for the CRFSuite library (Okazaki, 2007).

The features used in the linear chain CRF model do not create an adequate context representation. Adding features from neighboring sentences to the model can potentially improve it, however, it increases feature dimensionality rapidly, making it difficult to capture long-distance dependencies. Recurrent neural networks models provide a more elegant solution to this issue.

4.2. LSTM Model with CRF Layer

LSTM uses gated cells to preserve memory through multiple time steps. Therefore it is possible to incorporate contextual information. Neural networks also tend to process high-dimensional representations better than some of the more conventional statistical machine learning methods.
We add a CRF output layer to the model, which allows us to jointly model the output labels. The CRF layer essentially maximizes the label sequence score s, computed as follows:

$$s(y_{1:T}) = \sum_{t=2}^{T} \mathbf{A}[y_{t-1}, y_t] + \sum_{t=1}^{T} \mathbf{a}[y_t] \tag{2}$$

where $\mathbf{A}$ is the matrix of transition probabilities, and $\mathbf{a}$ represents the individual probability of a label. Dialogue turn representations are fed into a bidirectional LSTM (Bi-LSTM) layer. The output of the Bi-LSTM layer is concatenated with the speaker identity input, and then followed by two fully connected hidden layers.
The full model architecture is shown in Figure 1. The bottom row shows the sequence of input utterances, with sentence embeddings and speaker identities. 1 means the speaker is the same as the previous one, and 0 otherwise. Backward and forward LSTMs are shown in blue, followed by the concatenation layer in green. Yellow blocks are hidden layers. The top row is the CRF layer. In this example three output labels are shown (A = “Analysis”, DO = “Debugging operation”, ATV = “Algorithmic thinking / Variable”). Rightward arrows indicate that each label depends on its preceding label.

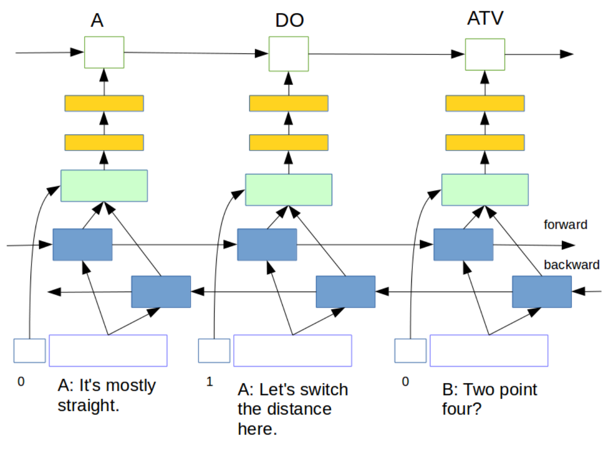

Figure 1: Bi-LSTM-CRF model.

We experimented different ways to obtain sentence encoders. (1) We use the same sentence encoder as in 4.1., so the sentence vectors fed into the Bi-LSTM layer are the same as before. (2) We train our own sentence encoder. For each sentence, pre-trained word embeddings are fed into a Bi-LSTM layer, followed by a Maxpooling layer. The encoder is co-trained with the classifier. (3) We combined the encoders from (1) and (2), concatenating the outputs of the two. In each case, we also supplement it with speaker identity.
Ideally the model should be time-variant and uses time step as variable too, because the time/stage of a dialogue may have an impact on the labels. However, currently we only have the full dialogues of two teams and there is no sufficient data to train a time-variant model.

4.3. Hyper-parameters

For the Linear chain CRF model, we set $c1 = 1.0$, $c2 = 10^{-3}$. These are L1 and L2 regularizations. We tried different number of $max\_iterations$ and found 200 to be desirable. All other hyper-parameters are just default values of the library.
For LSTM-CRF, our final model has 1024 LSTM units to process sequences of sentence embeddings, and the hidden layers have 1024 and 512 neurons, respectively. Input dropout for sentence embeddings is 0.4. The two hidden layers both have a dropout rate 0.5. When a sentence encoder is trained, 512 LSTM units are used. After Maxpooling, the results are concatenated with pre-trained sentence embeddings. The configuration of other parts of the neural network are intact.
There is no dropout for speaker identity input. All models are trained with 100 epochs. We found the results do not have noticeable changes when the epochs are set in the 80$\sim$200 range.
The glove.840B.300d word vectors[1] are used for word embeddings. It was trained on 840 billion tokens and represents tokens with 300d vectors.

4.4. Results

From 5-fold cross validation, the precision, recall and F1 scores of all 5 runs are calculated and averaged. Within each run, the scores are the prevalence-weighted macro-averages across classes. In other words, the scores are averaged over all classes and weighted by their frequencies. Using this method, recall will be the same as accuracy. It should also be noted that F1 score is not necessarily between precision and recall in this method. We calculated the standard deviation of F1 scores to check the stability of the results.
We list two naïve baselines to compare to. The majority baseline assumes a model assigns the majority class “O” to all occurrences. From the 5-fold CV, there are 5556 predicted labels and 3303 of them have “O” as the true label. Therefore the majority baseline is 3303/5556=0.594. The random baseline assumes a model assigns labels based on their statistical distributions in training data. Then the expected accuracy will be $\sum_i p_i^2$ where pi is the probability of class i. In our case the value is 0.384.
All the results are shown in Table 2. Our models outperform the two naïve baselines by a large margin. Judging from F1 score, the LSTM-CRF model with pre-trained sentence encoder has the best performance, although the recall

[1]https://nlp.stanford.edu/projects/glove/

Model	Prec	Rec/Acc	F1	F1 std
Majority	-	.594	-	
Random	-	.384	-	
CRF	.653	**.678**	.644	.0319
CRF+neighbor	.640	.675	.636	.0393
LSTM-CRF, pretrained	.660	.677	**.661**	.0241
LSTM-CRF, co-trained	.636	.654	.626	.0436
LSTM-CRF, combined	**.666**	.666	.654	**.0235**

Table 2: Evaluation results. "Majority" means assigning the majority label ("O" in our case) to all dialogue turns. "Random" means randomly assigning labels based on their statistical distributions. Precision, recall, and F1 scores are the averages of the 5-fold cross validation evaluation, respectively. "F1 std" is the standard deviation of the 5 F1 scores.

score is no better than that of the CRF model. LSTM-CRF with co-trained sentence encoder does not perform very well, probably because the training set is too small. Combining the two encoders does not boost results. It may suggest that the pre-trained sentence encoder is overall much better.

"CRF+neighbor" refers to the CRF model with neighboring sentence embeddings as input, in addition to current sentence. Such a technique incorporates more context in the model, but actually the performance is lower. Probably the data dimensionality has become too high for a CRF model to work properly.

Taking a close look, we find the CRF models tend to ignore low-frequency classes. For example, the smallest class QR ("query robot") occurs 37 times in all the 5556 test instances. The CRF model does not capture any of them (0 recall), but the LSTM-CRF model with pre-trained sentence embeddings manages to identify 1 of them correctly. The second smallest class QS ("query software") occurs 116 times. The CRF model only identifies 1, but the LSTM-CRF model finds 18 of them. Depending on the purpose, there can be more specific ways to evaluate the results, which will not be discussed in this paper.

5. Conclusion

In order to automatically label dialogue turns with computational thinking codes, we experimented with two computational approaches: (1) a linear chain Conditional Random Field (CRF) model, and (2) a recurrent neural network model with a CRF layer. Both of them beat naïve baselines by a large margin. Although our dataset is relatively small, the RNN-based model seems to outperform the conventional CRF model.

Our research is also a stepping-stone for microgenetic analysis of observational data. Eventually the analysis will help us understand collaborative learning and working in a team setup.

6. Bibliographical References

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Conneau, A., Schwenk, H., Barrault, L., and LeCun, Y. (2016). Very deep convolutional networks for natural language processing. *CoRR*, abs/1606.01781.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Dernoncourt, F., Lee, J. Y., and Szolovits, P. (2016). Neural networks for joint sentence classification in medical paper abstracts. *CoRR*, abs/1612.05251.

Dernoncourt, F., Lee, J. Y., Uzuner, Ã., and Szolovits, P. (2017). De-identification of patient notes with recurrent neural networks. *JAMIA*, 24(3):596–606.

Hassanzadeh, H., Groza, T., and Hunter, J. (2014). Identifying scientific artefacts in biomedical literature: The evidence based medicine use case. *Journal of Biomedical Informatics*, 49:159–170.

Hirohata, K., Okazaki, N., Ananiadou, S., and Ishizuka, M. (2008). Identifying sections in scientific abstracts using conditional random fields. In *In Proc. of the IJCNLP 2008*.

Huang, K.-C., Chiang, I.-J., Xiao, F., Liao, C.-C., Liu, C. C.-H., and Wong, J.-M. (2013). Pico element detection in medical text without metadata: Are first sentences enough? *Journal of Biomedical Informatics*, 46(5):940–946.

Huang, Z., Xu, W., and Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Kim, S. N., Cavedon, L., and Baldwin, T. (2010). Classifying dialogue acts in one-on-one live chats. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 862–871, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

Lin, J., Karakos, D., Demner-Fushman, D., and Khudanpur, S. (2006). Generative content models for structural analysis of medical abstracts. In *Proceedings of the Workshop on Linking Natural Language Processing and Biology: Towards Deeper Biological Literature Analysis*, BioNLP '06, pages 65–72, Stroudsburg, PA, USA. Association for Computational Linguistics.

Margolis, A., Livescu, K., and Ostendorf, M. (2010). Domain adaptation with unlabeled data for dialog act tagging. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, DANLP 2010, pages 45–52, Stroudsburg, PA, USA. Association for Computational Linguistics.

McKnight, L. and Srinivasan, P. (2003). Categorization of sentence types in medical abstracts. In *AMIA*. AMIA.

Moldovan, C., Rus, V., and Graesser, A. C. (). Automated speech act classification for online chat.

Okazaki, N. (2007). Crfsuite: a fast implementation of conditional random fields (crfs).

Samei, B., Li, H., Keshtkar, F., Rus, V., and Graesser, A. C. (2014). Context-based speech act classification in intelligent tutoring systems. In Stefan Trausan-Matu, et al., editors, *Intelligent Tutoring Systems*, volume 8474 of *Lecture Notes in Computer Science*, pages 236–241. Springer.

Siegler, R., (2006). *Microgenetic analysis of learning*, pages 464–510. John Wiley & Sons, Hoboken, NJ.

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer.

Sullivan, F. R., Adrion, W. R., and Keith, P. K. (2015). Microgenetic learning analytics: A computational approach to research on student learning. In *Paper presentation at the annual meeting of the American Educational Research Association*, Chicago, IL, April.

Tavafi, M., Mehdad, Y., Joty, S., Carenini, G., and Ng, R. (2013). Dialogue act recognition in synchronous and asynchronous conversations. In *Proceedings of the SIGDIAL 2013 Conference*, page 117–121, Metz, France, August. Association for Computational Linguistics, Association for Computational Linguistics.

Wertsch, J. (1991). *Voices of the mind: A sociocultural approach to mediated action*. Harvard University Press, Cambridge, MA.

Williams, A., Nangia, N., and Bowman, S. R. (2017). A broad-coverage challenge corpus for sentence understanding through inference. *CoRR*, abs/1704.05426.

Zhang, X., Zhao, J. J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In Corinna Cortes, et al., editors, *NIPS*, pages 649–657.