CHITCHAT: CHat with Interactively Trained Chatbots

Zhaoyu Wang, Jiaxing Xu, Peiran Yao

Department of Computer Science and Technology, Tsinghua University

Summary

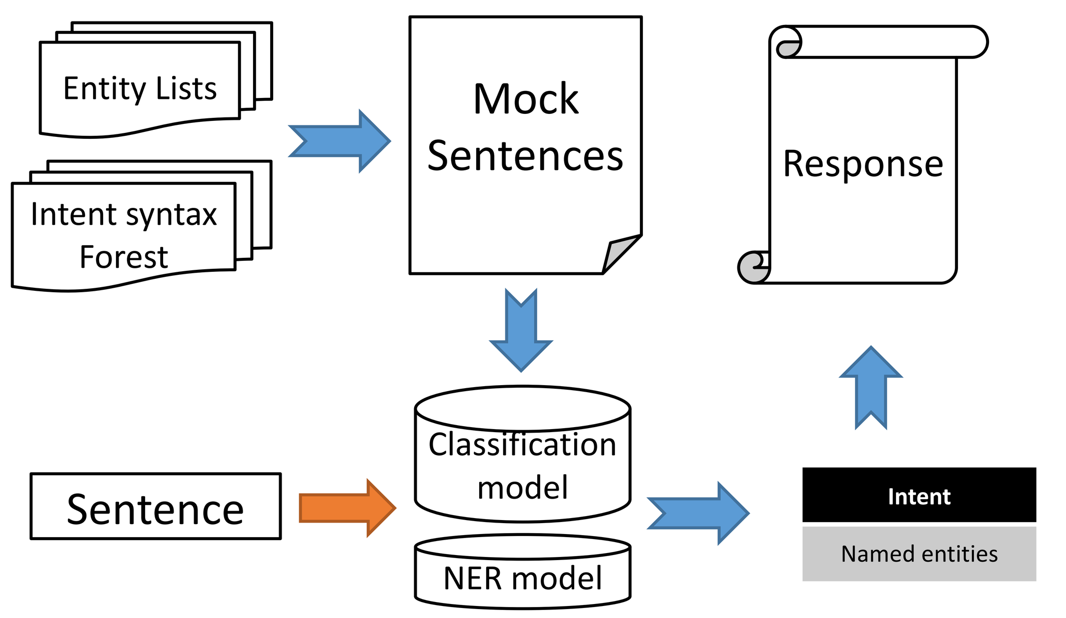
- Teaching machines to converse naturally with humans is challenging and really interesting.
- We propose a fantastic system to help people construct their own chatbots:
  - An interactive syntax tree help people to define question rules.
  - Use visualization methods to understand how to make mock sentences.
  - Use BLSTM-CRF-NER model and LSTM classifier to construct chatbot.

How it works

If you are a chatbot, once a user starts a dialogue, there are mainly three problems you want to understand clearly:

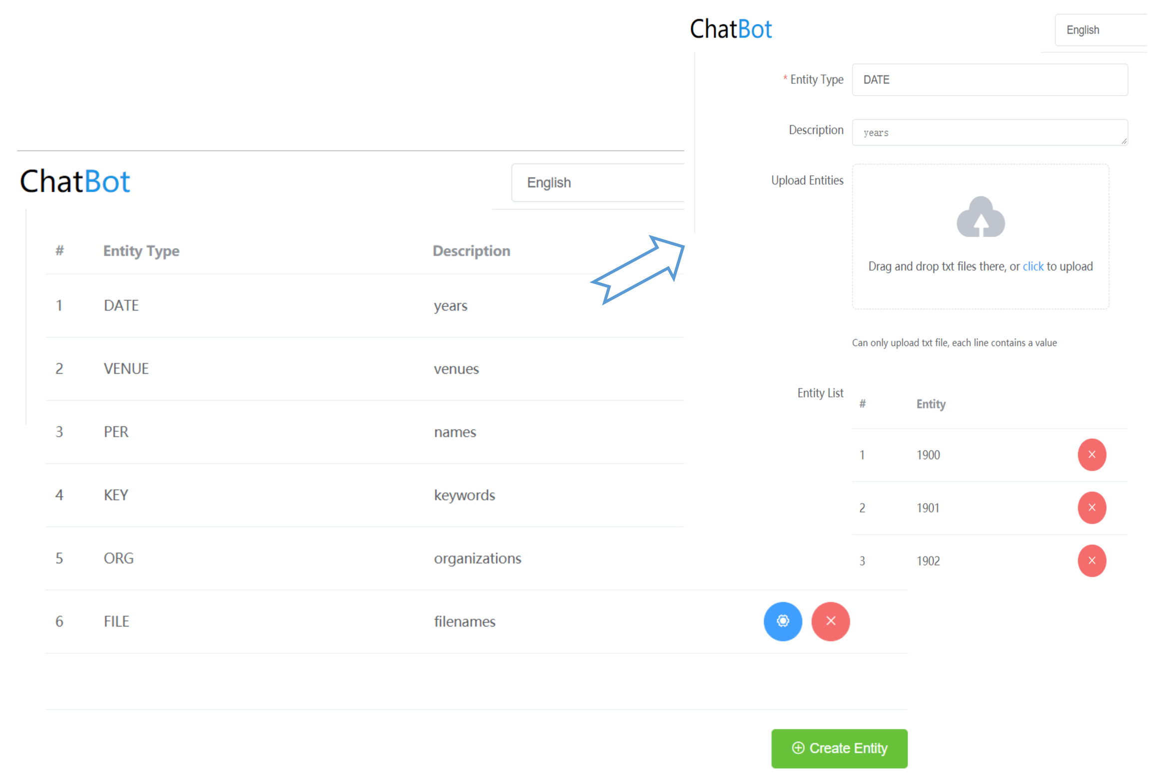
- What is the user's intent. Only if user's intent is accurately captured, you can make correctly actions or response.
- What is the key points (named entity recognizer, NER). If the user want to book a train ticket, you hope to confirm when the user want to go from where to where.
- How to response.

Thus there are many state-of-art classification methods and NER methods using neural network, which usually need many data to train model, we first need to generate some mock data. Then, we use these data to train classification model to capture the user's intent and train NER model to capture the key points of the sentence. Finally, based on those information, we could generate response for a given task.



Define Entity Lists

First we define some entity lists. An entity list is just like a word list. All you need to do is upload some words and give the entity name.



BLSTM-CRF-NER

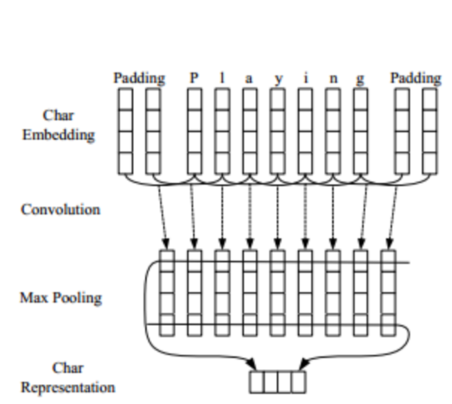
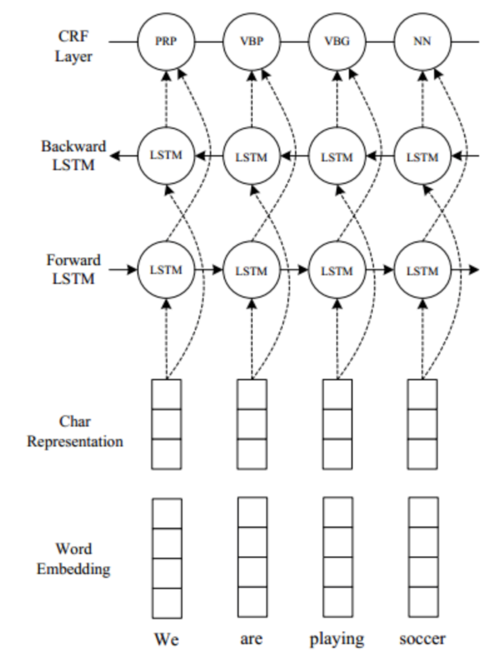


Figure 1: The convolution neural network for extracting character-level representations of words. Dashed arrows indicate a dropout layer applied before character embeddings are input to CNN.



Hyperparameter	Value
Tag scheme	IOB
Char embedding dimension	25
Char LSTM hidden layer size	25
Chinese word vectors	Wikipedia_zh Word+Ngram 300d
English word vectors	Wikipedia_en glove 300d
Char mode	char_LSTM
Epoch	20
Learning rate	0.015

Lample G, Ballesteros M, Subramanian S, et al. Neural architectures for named entity recognition[J]. arXiv preprint arXiv:1603.01360, 2016.
Ma X, Hovy E. End-to-end sequence labeling via bi-directional lstm-cnns-crf[J]. arXiv preprint arXiv:1603.01354, 2016.

```
processed 94643 tokens with 21171 phrases; found: 21170 phrases; correct: 21073.
accuracy:  99.82%; precision:  99.54%; recall:  99.54%; FB1:  99.54
            CON: precision:  99.92%; recall: 100.00%; FB1:  99.96  2478
           DATE: precision:  99.95%; recall: 100.00%; FB1:  99.98  2193
            KEY: precision:  99.50%; recall:  99.40%; FB1:  99.45  6842
            LOC: precision:  99.89%; recall: 100.00%; FB1:  99.94  1742
            ORG: precision:  99.12%; recall:  99.02%; FB1:  99.07  5583
            PER: precision:  99.61%; recall:  99.91%; FB1:  99.76  2332
```

https://github.com/AngusMonroe/BLSTM-CRF-NER

Return in json, Example:

```
http://166.111.5.228:5011/query/search some selection Thomas Edison State College 1902 Goel Shom 's papers

{
"LOC": ["selection", "1902"],
"PER": ["Goel"], "CON": ["Shom's"],
"DATE": [],
"ORG": ["Edison", "State"],
"KEY": ["Thomas", "College"],
"O": ["search", "some", "papers"]
}
```

Text Classification

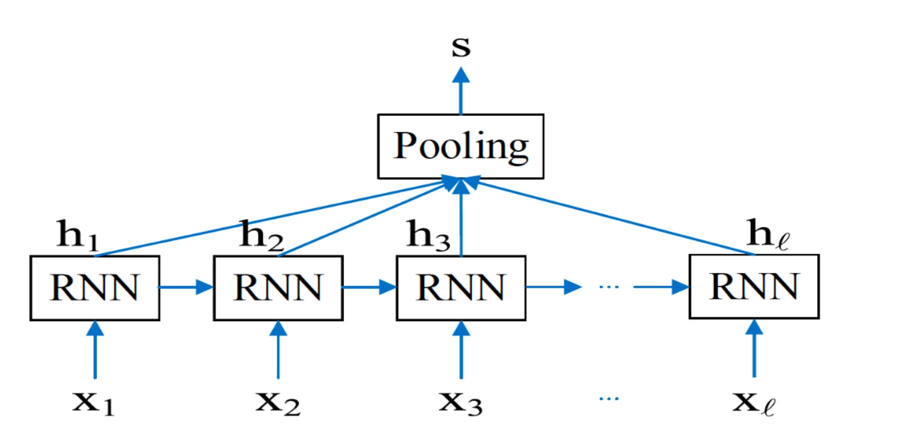


Figure 2-1: RNN architectures for generating the vector representation s of a short text $\mathbf{x}_{1:\ell}$.

Hyperparameter	Value
RNN	LSTM
RNN layers	2
Hidden size	128
Chinese word vectors	Wikipedia_zh Word+Ngram 300d
English word vectors	Wikipedia_en glove 300d
Epoch	100
Batch size	128
Learning rate	0.005

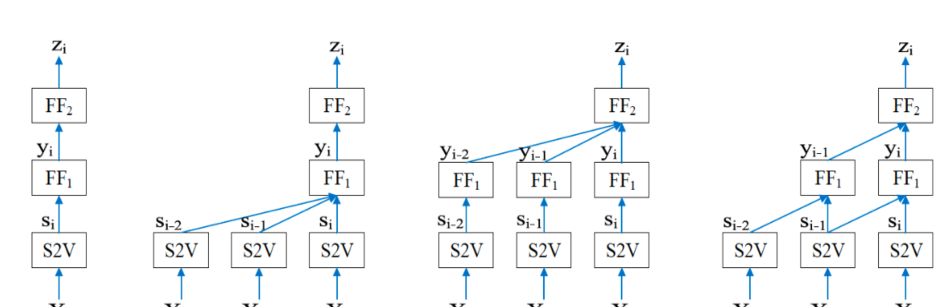


Figure 2-3: Four instances of the two-layer feedforward ANN used for predicting the probability distribution over the classes $\mathbf{z}_i$ for the i^{th} short-text $\mathbf{X}_i$. S2V stands for short text to vector, which is the RNN/CNN architecture that generates $\mathbf{s}_i$ from $\mathbf{X}_i$. From left to right, the history sizes (d_1, d_2) are $(0,0)$, $(2,0)$, $(0,2)$ and $(1,1)$. $(0,0)$ corresponds to the non-sequential classification case.

Phan X H, Nguyen L M, Horiguchi S. Learning to classify short and sparse text & web with hidden topics from large-scale data collections[C]//Proceedings of the 17th international conference on World Wide Web. ACM, 2008: 91-100.

```
Test: [10/156]  Time 0.117 (0.036)  Loss 0.0002 (0.0004)  Prec@1 100.000 (100.000)
Test: [20/156]  Time 0.026 (0.052)  Loss 0.0002 (0.0006)  Prec@1 100.000 (100.000)
Test: [30/156]  Time 0.019 (0.063)  Loss 0.0007 (0.0005)  Prec@1 100.000 (100.000)
Test: [40/156]  Time 0.017 (0.054)  Loss 0.0237 (0.0025)  Prec@1 99.219 (99.962)
Test: [50/156]  Time 0.046 (0.056)  Loss 0.0003 (0.0020)  Prec@1 100.000 (99.969)
Test: [60/156]  Time 0.041 (0.053)  Loss 0.0020 (0.0025)  Prec@1 100.000 (99.949)
Test: [70/156]  Time 0.039 (0.052)  Loss 0.0004 (0.0032)  Prec@1 100.000 (99.934)
Test: [80/156]  Time 0.031 (0.052)  Loss 0.0002 (0.0036)  Prec@1 100.000 (99.932)
Test: [90/156]  Time 0.022 (0.050)  Loss 0.0069 (0.0033)  Prec@1 99.219 (99.931)
Test: [100/156]  Time 0.030 (0.049)  Loss 0.0124 (0.0037)  Prec@1 99.219 (99.923)
Test: [110/156]  Time 0.129 (0.048)  Loss 0.0004 (0.0034)  Prec@1 100.000 (99.930)
Test: [120/156]  Time 0.018 (0.048)  Loss 0.0004 (0.0032)  Prec@1 100.000 (99.935)
Test: [130/156]  Time 0.020 (0.050)  Loss 0.0506 (0.0034)  Prec@1 99.219 (99.934)
Test: [140/156]  Time 0.017 (0.050)  Loss 0.0003 (0.0033)  Prec@1 100.000 (99.934)
Test: [150/156]  Time 0.066 (0.050)  Loss 0.0020 (0.0033)  Prec@1 100.000 (99.928)
 * Prec@1 99.925
```

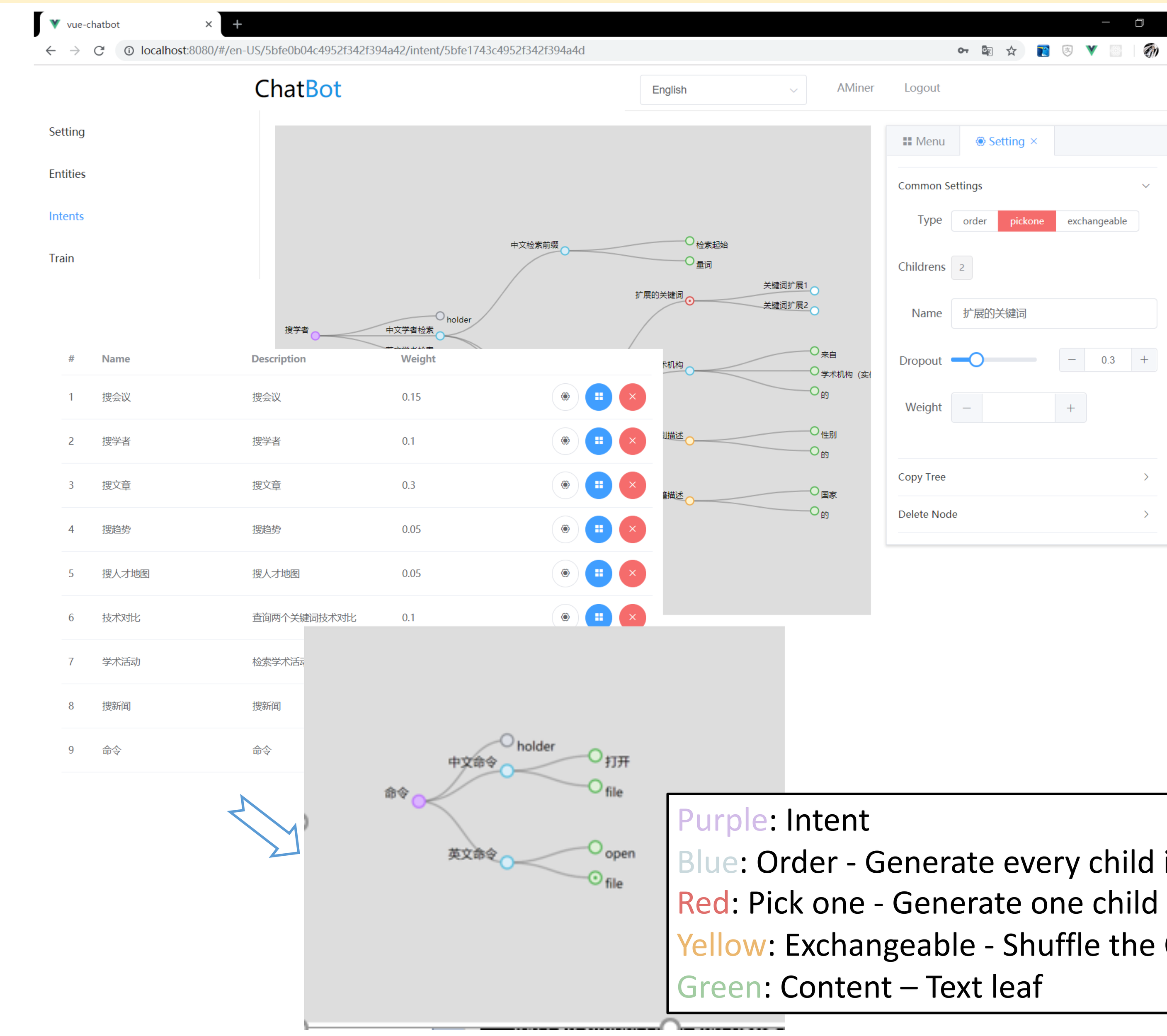
Return in json, Example:

```
http://166.111.5.228:5012/query/John
{"tag": "0"}
```

0-学者，1-文章，2-会议，3-chitchat

https://github.com/AngusMonroe/text-classification

Define Intent

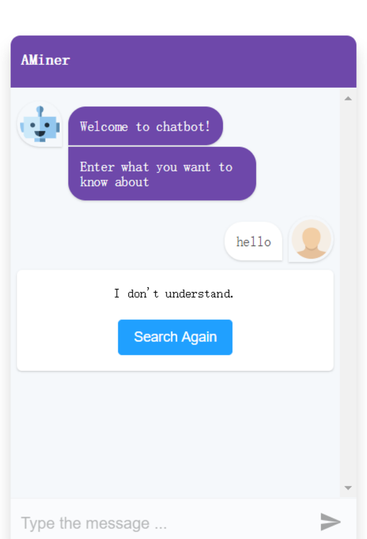


Purple: Intent
Blue: Order - Generate every child in order
Red: Pick one - Generate one child
Yellow: Exchangeable - Shuffle the Order result
Green: Content – Text leaf

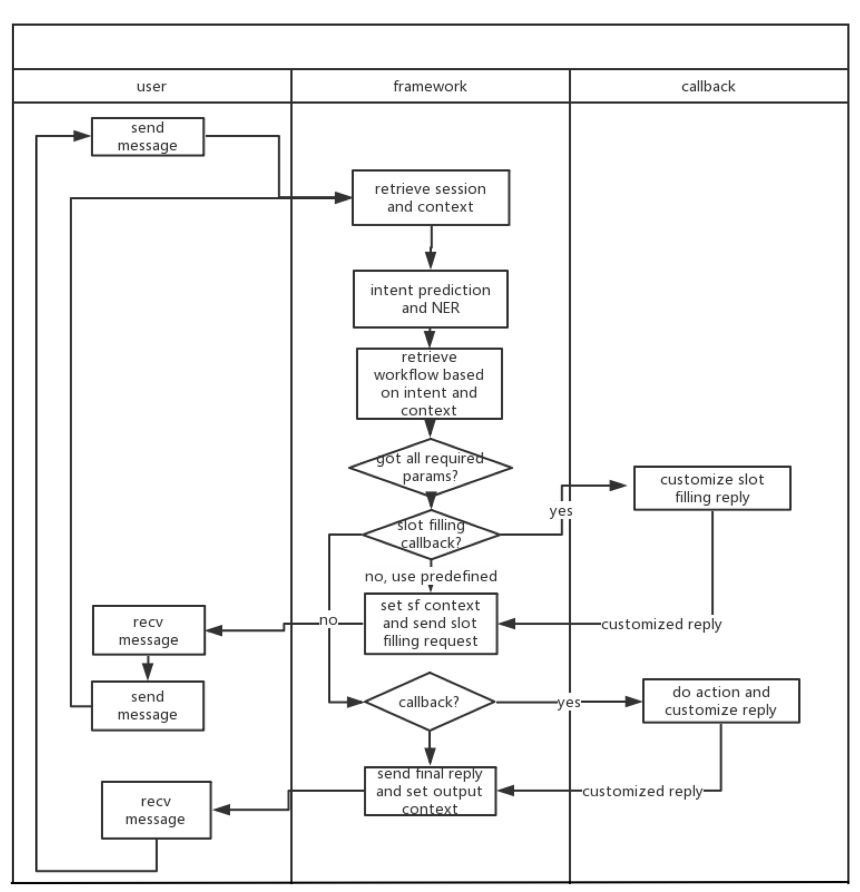
Chatbot Platform

You can easily create your own chatbots with our platform. Define entities and intents, the platform will do the following for you:

- train your model, with one click
- conversation API
- multi-turn dialog with context management



- identify and ask to fill slots
- predefined actions and replies
- customized callbacks
- embed to your own website



Future Work

- Combine active learning to optimize existing models.
- Encapsulate and reconstruct some code to enhance its robustness.