上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

自然语言处理 CS4314

中文口语语义理解

章俊一 520021910644

郑之胜 519141910033

张伟伦 520021910584

目录

1 前言

1.1 任务简介

口语语义理解是任务型对话系统中重要的一环，其目的在于从用户的话语中提取出用户的意图，为后续模块提供信息，最终顺利完成用户的任务。一般情况下，我们使用若干语义单元来描述用户的语义，每个语义单元由动作（act），语义槽（slot），槽值（value）三个部分构成。其中语义槽（slot）的类型限定了它可以被哪一类的槽值（value）所填充；而对话动作（act）侧重于对“对话行为”的表示，这一单元考虑了对话的轮次信息以及用行为来表达对话的意义，其中包括请求确认（confirm）、询问（request）等等行为。但是这种行为表示过于抽象，和更丰富具体的语义槽值信息是分离开的。因此，为了可以同时表达更具体的意思，我们采用一种简单有效的语义表达形式——对话语义动作，包括了一句话的行为特征以及其所带的若干简单的语义槽值对（slot-value pair），诸如 $act(slot) = value$ 这样的形式。

1.2 项目贡献

在本项目中，我们首先基于助教提供的基准方法 BiLSTM，对项目内容以及导航数据集的特点进行分析。基于分析结果，我们在原始方法的基础上进行了如下几方面的探索与改进。

- 在数据集方面，对于数据量不足的问题，我们设计了一种基于字符串替换的简单但有效的数据增广方式；对于 ASR 识别精度不足的问题，我们利用开源项目 pycorrector 对识别的文本进行修复。此外，除了助教提供的导航数据集，我们额外收集了两个中文口语理解数据集 CAIS 以及 ECDT。
- 在文本编码方面，我们探索了两种编码的方式——词编码以及字编码。在确定最终使用的方案为基于预训练语言模型的字编码后，我们又设计了两种处理中、英文文本同时出现时文本难以分词的方案。
- 在模型设计方面，我们首先将 baseline 使用的 BiLSTM 替换为目前最为流行的 Transformer，并且预测准确度上实现了大幅的提升。另外，我们使用在大规模中文文本上预训练的 BERT 以及 BERT 的相关变体来初始化模型的参数，使得模型收敛速度大大加快。最后，我们在 Transformer 的架构基础上，创新地设计了两种变体来解决 Tagging 预测的问题，分别是级联 Transformer 以及基于多个预测头的分类。
- 在实验方案方面，我们在四种不同的数据集上测试了四种模型设计方案，并且与目前的 State of the Art 的方法进行比较。实验结果证明，我们在两个数据集上相比于之前的方法有着显著提升。此外，我们还通过测试我们的方法在混合数据集上的表现，验证了我们的方法有着很好的泛化性。最后，我们设计了一系列的消融实验，以探索我们的方法中最为有效的模块。

2 数据预处理

2.1 数据集

导航数据集

导航数据集是助教提供的，它的 domain 是导航软件的语音对话信息。

其中的训练数据集 train.json 包括 3200 条数据，每条数据包含一段对话信息。一段对话信息中的每一句语句的信息都包含了它的语音识别语句（asr_1best）、人工转义语句（manual_transcript）和它的语义信息（semantic）。其中 semantic 包含若干个语义单元，每个语义单元由动作（act），语义槽（slot），槽值（value）三个部分构成。

数据集中的 ontology.json 文件定义了该数据集所在 domain 包括的动作（act）的类型和语义槽（slot）的类型和部分槽值。其中动作只有肯定'inform' 和拒绝'deny' 两种。地点类、操作类和序号类语义槽的槽值则被存储在 lexicon 目录下。

导航数据集还提供了与训练数据集结构相同的开发数据集 development.json 以供模型测试使用。development.json 包含 527 条数据。另外还提供了一个没有经过人工转义，也没有打上语义信息 semantic 标签的测试样例数据集。

CAIS

CAIS 数据集 [3] 是从MLWA-Chinese-SLU项目提供的原数据集中转换而来，它的 domain 是智能音乐播放器的客户语音识别信息。

原 CAIS 数据集有五个子数据集，包括四个使用了不同的分词器进行分词的子数据集和一个由人工正确分词的子数据集。每个子数据集都包括了训练数据集、开发数据集和测试数据集，它们结构相同。以训练数据集为例，它包含一个分词数据集文件和一个以字为单位的序列标注文件。分词数据集文件的每个单元中存储了一条语句和语句的分词结果。序列标注文件的每个单元中存储了以字为单位的语义单元和句子的动作语义类别，每个语义单元由字、BIO 标注信息、分词信息构成。其中若某一个词属于 CAIS 预定义的词语义类别（如歌手名、曲目名等），则 BIO 标注信息还将附加词语义类别信息。而句子的动作语义信息对应的是整个句子的语义类别。

我们对比原 CAIS 数据集和导航数据集后发现：句子的动作语义信息与导航数据集的动作信息相似，BIO 标注附加的词语义类别信息与语义槽相似，而有附加信息的词块则是相应的槽值。因此我们依照这个映射关系将原 CAIS 数据集转换成符合我们项目的数据集。

ECDT

ECDT 数据集 [3] 也是从MLWA-Chinese-SLU项目提供的原数据集中转换而来，它的 domain 是手机智能语音助手识别到的用户语音信息。它的原数据集结构与 CAIS 数据集相似，因此我们以相同的方式对原数据进行了转换。另外，我们观察到 ECDT 数据集中存在更多的与导航相关的信息，因而它的 domain 比 CAIS 更接近于导航数据集。

2.2 数据集扩充

在导航数据集的训练数据中，我们观察到：若转义语句中存在已定义的槽值 value，则会被打上相应的语义槽 slot 标签。而一种 slot 标签对应若干个值 value，并且是可以被互相替换的。例如: “导航到凯里大十字” 这个语句中，由于凯里大十字属于语义槽 “终点名称”，而终点名称对应的语义槽值中还有诸如 “丽江古城”、“乌当中学” 等地名可以被替换进原句中并且合乎逻辑。

因此我们的扩充方案就是：首先选择一个语句中的一个语义槽，然后在该语义槽类别对应的槽值中随机选取若干条信息替换生成新语句作为新数据集。表1是几个扩充样例。

表 1: 数据集扩充样例

原句	语义槽	原槽值	新句	新槽值
导航到凯里大十字	终点名称	凯里大十字	导航到大桥公园	大桥公园
导航到凯里大十字	操作	导航	返回到凯里大十字	返回
关闭导航	操作	关闭	打开导航	打开
关闭导航	对象	导航	打开高德地图	高德地图
查找附近的酒店	请求类型	附近	查找近郊的酒店	近郊
导航到济南西收费站不走高速	路线偏好	不走高速	导航到济南西收费站高速优先	高速优先

在扩充方案里，我们没有将一个语义槽对应的槽值全部替换生成新句。这样做的目的是为了防止数据过拟合。原因是在导航数据集中，地址类语义槽值的数目远远大于其他类语义槽的槽值数。如果将地址类语义槽全部替换将会使数据集中地址类的特征进一步加强，出现过拟合。

2.3 ASR 恢复

在导航数据集中，我们观察到：部分语音识别语句 ASR 存在谐音错误、混淆音错误和形似字错误。对此我们做了简单的数据分析，发现原训练数据集中 semantic 中的槽值在 ASR 中不存在的比例高达 11.44%。因此我们认为应该尝试在读入 ASR 后，首先对 ASR 进行错误修正。

我们采用的修正工具是中文文本纠错工具pycorrector，它的原理是：

1. 中文纠错分为两步走，第一步是错误检测，第二步是错误纠正；

1. 错误检测部分先通过结巴中文分词器切词，由于句子中含有错别字，所以切词结果往往会有切分错误的情况，这样从字粒度和词粒度两方面检测错误，整合这两种粒度的疑似错误结果，形成疑似错误位置候选集；

2. 错误纠正部分，是遍历所有的疑似错误位置，并使用音似、形似词典替换错误位置的词，然后通过语言模型计算句子困惑度，对所有候选集结果比较并排序，得到最优纠正词。

3 数据分析

3.1 对话历史分析

在任务简介1.1 章节中我们可知，对话动作（act）更加偏重于“对话行为”的表示，适合于多轮对话的情景，而槽值和语义槽信息则不能直接与前后的对话历史产生关联。因此，如果训练数据中涉及丰富的多轮对话历史且其中包含多样的对话动作的话，对话历史或许会对最后的语义理解有较大帮助。但遗憾的是，当前的训练样本中，只涉及两种动作，分别是 inform 和 deny，且其中 deny 的条数非常少，几乎不能够拿来分析的依据。

另外，在对数据集逐条检查之后，发现多轮的对话样本中每轮对话相互独立，tagging prediction 所需信息完全可以由单条对话判断出来；由于本任务涉及的 domain 是导航类别，几乎不存在 domain shift 的问题，因此从这个角度来分析也不需要多余的对话历史来辅助。

3.2 ASR 结果分析

在2.3中我们提到，可以尝试对 ASR 中的错误数据进行恢复。我们使用的修正工具是 pycorrector。下表是部分 ASR 恢复后的句子示例。

表 2: ASR 恢复样例

原句	新句
一口客运站	义口客运站
放中医院	方医院
我要去退太原市	我要去北太原市
我要去卸店街	我要去饭店街
我要去紫金县城	我要去紫明县城
导航去桐昆集团	导航区桐昆集团
到独的路派出所	到独得路派出所
下灯线九江煤矿	下申线九江煤矿

观察样例可以发现，被 ASR 修正的大多是地址类槽值。地址类槽值的特点便是与 domain 强烈相关，pycorrector 模型完全没有见过的地址名几乎不可能被正确修正。因此出现了原本正确的“去桐昆集团”被错误修改成了“区桐昆集团”的例子。可见，对于导航数据集这样 domain 比较独特，存在许多无法泛化的地址名称的情况，用文本修正工具的意义不大，帮助有限。

4 词向量

词向量（Word embedding）是一组语言建模和特征学习技术的统称，它涉及从每个单词一维的空间到具有更低维度的连续向量空间的数学嵌入。而当下主流的获得词向量的方式便是 word2vec 算法，word2vec 使用神经网络模型从大型文本语料库中学习单词关联，这样的模型经过充分的训练后便可以为每个不同的颗粒度的单元生成独特的数字向量空间。在这一章节，将介绍若干种不同颗粒度的词向量表征形式。

在中文的 NLP 任务中，使用场景最多的便是字向量。[2] 通过大量的直接对比实验证明了在中文的 NLP 任务中，字向量普遍比词向量要更好，这是因为以词为单元的模型很容易受困于数据的稀疏性与 out-of-vocabulary（OOV），进而更容易会过拟合。

4.1 字向量

字向量，即模型处理的文本单元是字，这种形式的向量不需要对中文语句进行分割，直接可用。在本任务中，我们尝试了默认的词向量、来自北京师范大学和人民大学推出的 Chinese-Word-Vectors 的不同语料场景下的中文向量和腾讯 AI Lab 提供的预训练向量表征Embedding Datasets。

4.2 词向量

中文中以词作为向量的文本输入单元的时候，需要对输入的完整的语句进行分割，切分成一个一个的语义单元，即语义独立的词。在本任务中，我们尝试用 jieba 这个工具包进行分词，进而基于分词后的结果作为文本单元输入到模型中。其中，使用的向量表征依旧来源于上文的 Chinese-Word-Vectors。对于 OOV 的词汇我们将其认定为是 UNK。从结果上来看，以词为单元的处理形式在当前如此小量的训练文本下非常容易过拟合，从而导致在验证集上泛化性能很差，acc 很低。

5 模型架构优化

在这个章节，我们首先介绍 baseline 所采用的方法 BiLSTM，接着我们介绍在本工作中利用的模型 Transformer，以及我们设计的 Transformer 的两种变体，级联 Transformer (Cascaded Transformer)，以及多分类头标签预测 Transformer (Multi-Head Tagging Prediction Transformer)。

5.1 BiLSTM

BiLSTM 或双向 LSTM 是一种序列处理模型，由两个 LSTM 组成：一个向前接收输入，另一个向后接收输入。BiLSTM 有效地增加了网络可用的信息量，改进了 LSTM 算法可用的上下文（例如，知道什么词紧跟在句子中的某个词之前和之后）。因此，在训练过程中模型的感受野是整个句子，可以充分的挖掘句子中潜在的上下文语义信息。

将 BiLSTM 作为整个架构的 backbone, 进行 BIO 序列标注。具体而言，该序列标注为当前对话中的每个符号（词或字）打上一个标签。标签可能为 “B-X”、“I-X” 或者 “O”。其中，“B-X” 表示该符号所在的

片段属于 X 类型并且此符号在此片段的开头，“I-X”表示该符号所在的片段属于 X 类型并且此符号在此片段的中间位置，“O”表示不属于任何类型；其中“X”则是上文所讲述的 slot-value pair。从某种程度上来说，在进行这样的处理之后，任务转换成经典的分类任务，将模型的预测输出与真实的标签做交叉熵便可进行 loss 的计算和后续的训练。下文中所涉及的不同模型只是调整架构中的 backbone, 其他部分与此章节内容相同，故不再赘述。

5.2 Transformer

Transformer

Transformer 是一种神经网络架构，最早由 Google Brain 在 2017 年的论文"Attention Is All You Need" [4] 中提出。Transformer 架构的关键创新在于使用了自注意机制，这使得模型在做出预测时能够衡量输入不同部分的重要性。

Transformer 架构的一个主要优点是能够有效地处理顺序数据，如文本或时间序列数据。这是因为自注意机制允许模型在做出预测时考虑整个输入序列，而不是像传统 RNN 架构那样只考虑之前的时间步。

Transformer 架构由编码器和解码器组成，两者都由多层自注意和前馈神经网络组成。编码器输入序列并产生一组隐藏状态，然后传递给解码器以产生输出序列。

BERT

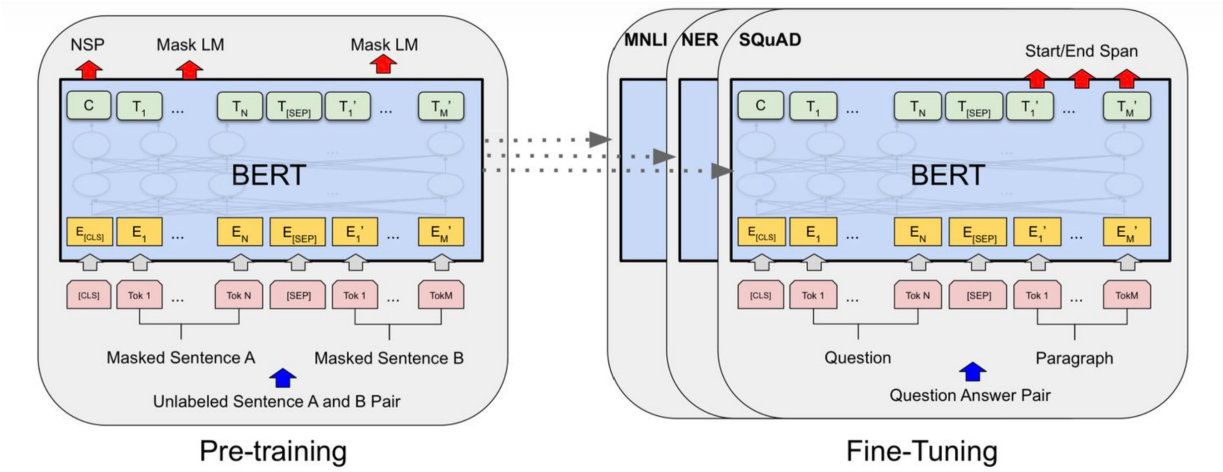

图 1: BERT 的模型示意图

BERT（Bidirectional Encoder Representations from Transformers）[1] 是一种预训练方法，用于基于 Transformer 的模型，由 Google 于 2020 年提出。BERT 是在大量文本数据上使用预训练任务训练的，称为掩码语言建模，模型被训练用来预测句子中缺失的单词。这个预训练步骤使得模型能够学习输入文本的丰富表示，然后可以用于各种自然语言理解任务，如情感分析、命名实体识别和问答。

预训练 BERT

在我们的工作中, 我们利用了中文预训练 BERT 来初始化模型的权重, 可以加快模型的训练速度。因为预训练好的 BERT 模型已经从大量文本数据中学习了很多常规知识，所以在较小的特定任务数据集上进行微调时可以使用更少的数据。

5.3 Cascaded Transformer

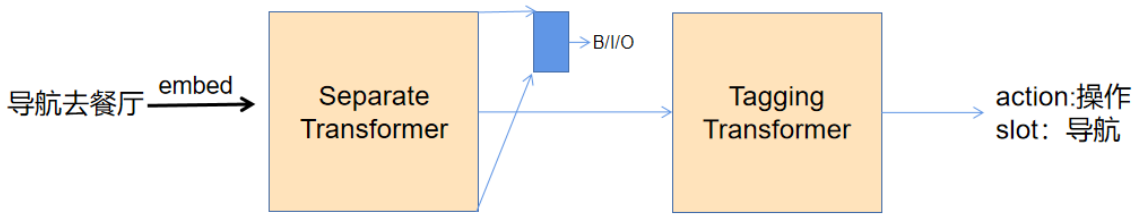

图 2: Cascaded Transformer 的模型示意图

Cascaded Transformer 是 Transformer 架构的一种变体，其中多个 Transformer 被级联在一起形成更深的网络。与单个 Transformer 相比，Cascaded Transformer 使用了多个 Transformer 级联在一起来进行序列标注任务。在我们的工作中，我们在原版 Transformer 的基础上进行了改进，使用第一个 Transformer 的隐藏状态来预测 O/B/I tagging, 并使用第二个 Transformer 来预测完整的 tag。

这样做的好处在于，第一个 Transformer 可以捕捉词语与词语之间的关系并产生对应的隐藏状态，然后第二个 Transformer 可以利用这些隐藏状态来进行完整标注。这样可以更好地捕捉序列中词语与词语之间的关系，从而提高标注的准确性。

5.4 Multi-Head Tagging Prediction Transformer

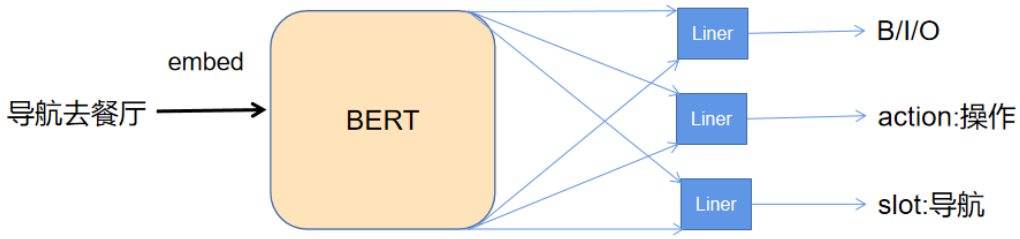

图 3: Multi-Head Tagging Prediction Transformer 的模型示意图

Multi Head Tagging 是一种在 Transformer 架构上使用多个头来进行序列标注任务的方法。在我们的工作中，我们在 Transformer 最后一层隐藏状态上使用三个分类头来分别预测 tag 的 O/B/I，action 以及 slot。这样做相比于之前预测完整的 tag（即同时包含 O/B/I，action 以及 slot 三者）有很多优点。

首先，使用多个头可以更好地捕捉输入序列中不同部分之间的关系。其次，使用多个头可以更好地利用输入序列中的信息，因为每个头都可以学习不同的表示。另外，使用多个头可以提高标注的准确性，因为每个头都可以预测不同的标签。最后，使用多头分类比使用单头分类能够节省更多的参数量。这是因为，在使用一层线性层的情况下，多头分类只需要使用 $\text{hidden\_size} * (4 + l + n)$ 的参数量，而单头分类则需要

使用 hidden_size $*(4*l*n)$ 的参数量，其中 l 为 action 的标签数量，n 为 slot 的标签数量。这样的设计可以使得模型更加高效地利用参数。

6 实验设计

6.1 实现细节

我们的所有实验都在单张 RTX3090 GPU 上进行，模型训练过程使用 AdamW 优化器进行优化，学习率在 {1e-5, 3e-5, 5e-5} 中搜索最佳的参数，模型使用的是 BERT [1] (base 版本)，模型初始化参数在 {"bert-base-chinese", "chinese-macbert-base", "chinese-lert-base", "chinese-pert-base", "chinese-roberta-wwm-ext", "hfl/chinese-bert-wwm-ext"} 中搜索最佳的预训练模型。

对于导航数据集，我们在训练时分别尝试了仅在 asr 数据上、仅在标注数据上、在 asr 以及标注的混合数据上进行训练，并在测试时分别测试了 development set 在 asr 数据上以及在标注数据上槽值的准确率。

对于将预训练 BERT 的 tokenizer 作用在中英文混合的文本数据上，并且使得其按字符分词，我们设计了两种方案。第一种方案是，对于无法按照字符分词的英文单词，我们将每个非中文字符通过空格分割，作为新的单词。第二种方案是，对于上述提到的单词，我们首先获得每个单词 tokenizer 后的 id，再通过将 id 重复填充的方式，补充到原始字符长度。

6.2 不同的初始词向量的表现

如表3 所示，采用相同的训练数据、相同的 bsaeline（BiLSTM）模型架构，一致使用人工标注作为文本输入，可以看到使用不同的词向量模型的收敛轮数存在较大的不同，且最终的标价指标 acc 也不尽相同。

表 3: 不同初始词向量对训练结果的影响表

词向量	收敛轮数	acc
word2vec-768	87	93.18
sgns.zhihu.word	48	94.19
sgns.zhihu.bigram-char	40	93.97
sgns.context.word-word.dynwin5.thr10.neg5.dim300.iter5	8	93.07

6.3 单个数据集上的表现

表 4: 不同模型架构在特定数据上的定性结果

方法	导航数据集（ASR/标注）	CAIS 数据集	ECDT 数据集
Stack-Propagation	-	84.68	31.56
Slot-Gated	-	80.83	26.72
SF-ID Network	-	82.41	25.07
MLWA-Chinese-SLU	-	86.17	34.08
BiLSTM	71.40/93.18	83.61	68.09
Transformer	78.32/95.23	86.62	72.03
Cascaded Transformer	78.66/94.30	87.93	73.64
Multi-Head Tagging Prediction	79.22/95.64	87.53	77.00

在这个部分，我们测试本项目中所用到的四个方法在三个数据集上的表现，我们利用模型在 development set 上的预测准确性来验证方法的有效性。我们与 MLWA [3] 文中提到的四个 baseline 进行比较。

从表 4中可以看出，在本项目的四个方法中，我们设计的基于 Transformer 的两种变体 Cascaded Transformer 以及 Multi-Head Tagging Prediction 相比另外两种方法均有显著的提升，体现了我们的方法的优越性。而与 MLWA [3] 一文中的 baselines 相比，我们设计的两种变体也相比 SOTA 的方法有进一步的提升，其中 ECDT 数据集由于没有公开的测试集，因此可能有一定的区别。

6.4 跨数据集上的表现

表 5: 不同模型架构在混合数据集上的定性结果

方法	导航数据集（ASR）	CAIS 数据集	ECDT 数据集	混合数据集
Transformer	77.21	84.61	74.67	82.47
Multi-Head Tagging Prediction	77.54	82.50	75.45	82.78

在这个部分，我们通过判断模型在混合数据集上的表现，来验证方法的泛化性。我们在训练时，使用导航数据集（ASR 结果）、CAIS 数据集以及 ECDT 数据集的混合数据进行训练。在测试时，我们分别在导航数据集（ASR）、CAIS 数据集、ECDT 数据集以及三个数据集混合的 development set 上测试预测的准确度。

从表5中可以看出，得益于对 tagging 预测的功能拆分（拆分成 B/I/O，action，slot 三部分），Multi-Head Tagging Pred 相比传统的 Transformer 在四个数据集上都有着更好的表现，说明了这样的拆分能够提高模型在多个数据集上的泛化性。

6.5 消融实验

表 6: Multi-Head Tagging Prediction 在导航数据集上的消融实验

方法	导航数据集（ASR/标注）
w/o 数据增广	-/95.08
tokenizer v1	78.54/95.33
with pycorrector	76.76/-
all	79.22/95.64

从表6中可以看出，数据增广对于提高标注数据上的预测准确性有很大的帮助；考虑了英文单词含义的 tokenizer v2 相比 tokenizer v1 也有着更好的表现；pycorrector 在某些情况下，可能会将正确的数据修改成错误的结果，所以导致最终结果较差。

7 项目总结

在本项目中，我们基于原始 BiLSTM 方法在**数据集**、**文本编码**、**模型设计以及实验方案**四个方面均进行了较为深入的探索与改进，并且最终通过在 development set 上的测试，验证了我们的改进方案优于 BiLSTM 以及 SOTA 的方法。当然，限于时间原因，本项目未来还有很多提升空间。例如，如果有更多的文本训练数据，以词为单元的向量或许是一个可以考虑的方案。以及，可以通过将语义槽的文本编码纳入模型预测，提升 Out of Domain 预测问题的表现。

参考文献

[1] Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers

for language understanding. *CoRR abs/1810.04805* (2018).

[2] Li, X., Meng, Y., Sun, X., Han, Q., Yuan, A., and Li, J. Is word segmentation necessary for deep learning of Chinese representations? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, 2019), Association for Computational Linguistics, pp. 3242--3252.

[3] Teng, D., Qin, L., Che, W., Zhao, S., and Liu, T. Injecting word information with multi-level word adapter for chinese spoken language understanding. *CoRR abs/2010.03903* (2020).

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *CoRR abs/1706.03762* (2017).