# Summary

# DSTC1: Linear Combination of Scores

## ROC.V2\_05 Result (run 2 days):

All zeros.

Conclusion, the “3way” method doesn’t do well on ROC.V2\_05.

# New Challenge: DSTC2

# Challenge Topline

* Topline1: If the correct one appears in one of the ***SLU***, add it to the output; if it doesn’t appear, ignore it.
* Topline2: If the correct one appears in ***ASR***, it is correct
* Topline3: If the correct one appears in ***Transcription***, it is correct

For the “topline2” and “topline3”, only the results with all candidates are reported because there is only one transcription and the ASR is not sorted as the NLU. In other words, the top NLU might not come from the top ASR.

## Results

## Observation

### “joint goals”

* For the Topline3 > Topline2 > Topline1 > 2wayTopline>HWUBaseline>Baseline
* Only considering the top1 is not enough. It is because the HWUbaseline is very close to topline1 with top1 and it will be really hard to beat it.
* It is possible to improve the performance by considering N-Best. For topline1, the performance for top10 is 84.9% compared to 74.6% for top1. It can get to 80.7% even by considering just top2.
* ASR error is a big issue. The result for “topline3” is much better than topline2. The best one is 95.8% based on the transcription. [It is not 100% because of some of slot-value cannot be found in the transcription, such as “pricerange = moderate”, “area = dontcare”]

### “method” & “request”

The “method” and “request” are inferred by the SLU, so that their topline2 and topline3 results are the same as topline1. [ASR and Transcription don’t help. For example, for “request=phone”, “phone” might not appear in the transcription.]

However, we can treat them as an independent problem by building classifiers for them. For “method”, it is just a 5-way classification problem [The only issue is that the current result depends on the previous one. For example, if there is no method mentioned at the current turn, the method is assumed as same as the old one]; for “request”, only 8 things are “requestable”, so we can build a classifier for each one by asking “Is address requested?”, “Is phone requested?”, etc.

# H2

*Assume that the SLU is correct if and only if it matches the differences between the pervious correct answer and the current one*

# H3

*Assume that the SLU is correct if and only if it will turn into the correct answer based on the previous one and the new SLU*

## Rank Distribution

With different hypothesis, the rank distribution is different, which is shown in the table below.



## H1, H2, H3 Topline

## Observation:

* H1 > H3 > H2
* This result surprised me first. However, it makes sense. The reason is that H1 doesn’t rely on the previous turn and H2 and H3 assume the previous one is correct. However, the previous one might not be right.

For example,

Turn 1:

System: “dojo noodle bar is a nice restaurant in the centre of town serving asian oriental food”

User: phone number

NLU: []

Annotation: request = phone

Turn 2:

System: dojo noodle bar is a nice restaurant in the centre of town serving asian oriental food

User: phone number

NLU: request = phone

Annotation: request = phone

For H2, the difference between Turn 1 and Turn 2 are none, so the NLU in Turn 2 will not be accepted. However, since the NLU recognizes nothing in Turn 1, it should accept the second chance.

# Method Classification

Assume the “method” is only depended on the current turn, then it becomes a simple 5-way classification problem: [none,byconstraints,byname,finished,byalternatives]

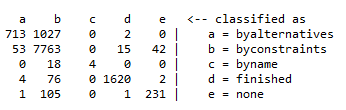
## 5-way result

Model: SVM

Feature: Act + Unigram of the ASR

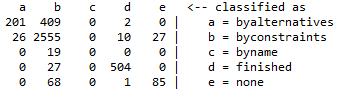
|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F-Measure |
| dstc2\_train | 0.891 | 0.885 | 0.869 |
| dstc2\_dev | 0.851 | 0.85 | 0.826 |

Confusion Matrix for training set:



The accuracy is (713+7763+4+1620+231)/11677=88.5%, which is not better than the 2way model (89%)

The Confusion Matrix for dev set:



The accuracy is (201+2555+504+85)/3934=85.0%, which is less than 2waymodel (86.4%)

However, the problem is, the method is highly depended on the previous turn. If there is no information for the current turn, the method should be the same as the old one. In this case, it might be better to include the previous label as a feature. However, for the test set, the previous label should be predicted first and then can be used as a feature. This is TODO.

# Considering top1 + top2

TODO:

# Request Classification

TODO: