Cognitive Approach to Natural Language Processing

Setting the value of necessity based on a real dialogue

Qingjie Zhang and Temur Malishava

Problem statement (Qingjie)

As we have seen in the lab of argumentation, we can generate relevant arguments and do deliberative reasoning, from a real dialogue, with CAN (Conflict-Abduction-Negation) procedure, in which the notion of necessity subsumes desires and beliefs. However, in the lab the necessity of a certain affair is a priori setted. What's more, during the lecture we have not learned how its absolute value is determined but only the sign (if an affair is willing to happen then it is positive otherwise it is negative). In the example of the lab, it is logical that the absolute value of "tough_work" is greater than that of "nice_doors" because the workers don't want to repaint the doors in a tough way. But how do we know it from the original dialogue? What's more, how does the machine know?

In this mini project, we will study the absolute value of necessity of an affair in the real dialogue. And we will try to let the machine know the necessities on the fly, rather than setted a priori.

Relevant studies (Qingjie)

Generally, necessity corresponds to multi-valued logics and unifies beliefs and desires. In the belief–desire–intention software model (BDI) [1], necessity is explained as a mechanism for separating the activity of selecting a plan (from a plan library or an external planner application) from the execution of currently active plans. In preference learning [2], a relation order, which we can identify as the order of the absolute value of necessity, on a collection of alternatives is predicted.

The models above may be too sophisticated to analyze a simple dialogue. In the following report, we propose a simpler model to extract the value of necessity.

Methods (Qingjie and Temur)

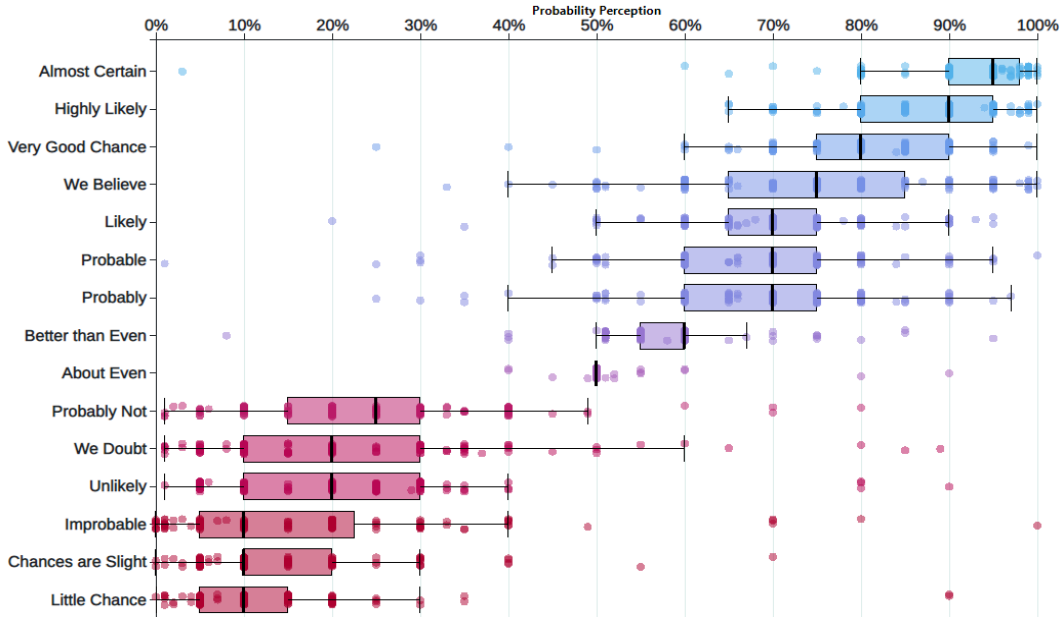
We notice that in real dialogue, the intensity with which a given situation is believed or wished can be delivered by certain words. For example, when we say "really", our intention is strong. When we say "whatever", we are indifferent. In short, our model is to identify these words indicating the necessity.

Perception of Probability Words (Temur)

Before we start discussing our methods in detail, first we have to think of an acceptable solution to correctly identify the relative necessity of each word. For that we have to understand the meaning of probability in language. In general, we know that the purpose of probability is about estimating how likely something is to happen. It can be used to estimate the likelihood of an outcome, for example, when throwing a die or tossing a coin.

Such probabilities can be described in words. With the help of language, we often make judgments as to whether an event will take place or not. For example, we might say that it is *likely* to rain tomorrow, or that it is *impossible* to meet on Monday because of the workload. Other words that come to mind are: *certainly*, *likely*, *probably*, *unlikely*, *impossible* and many others.

To rank these words accordingly we use the probability scale.



This image shows a result of a small survey about the perception of some probabilistic words [3]. In general, the word probabilities on the scale are written as fractions or decimals from 0 to 1, or in our case, as percentages from 0% to 100%. For example, as we can see a word *likely* for most people is in the 65-75% range of probability.

With the help of this study we can measure the necessity of each of the suggested words relative to each other in such a way that is accepted by the perception of the individual responses.

Basically, our suggestion is to set the necessity to 0 when the perception of a given word is at 50%. So, the necessity for the word *even* would be 0. For the words above 50% the

necessity will grow accordingly whilst for the words below 50% the necessity will drop. So for the word *likely* we will have necessity equal to 20 since it's in the 70% range whereas for the word *unlikely* we will have necessity equal to -30 since it's in the 20% mark.

How to extract preference from a dialogue and set it on the fly (Qingjie and Temur)

Following the previous finding, we create a table of the necessities of probability words. Let's look into this example:
"
A1-Do you want to hang out tomorrow?
B1-I'm sorry but it's unlikely that I will be able to. What about the day after?
A2-Yeah, probably, though I doubt it. But if we went on the weekend I would absolutely be free.
B2-Yes! I am free on the weekend as well. We can certainly meet up then.
"
We have marked the probability words in red. Here we can clearly follow the probability scale mentioned before and assign the necessities accordingly.

On the other hand let's consider a real dialogue like this:
"
Son: Oh! It's raining heavily! But I have to go to school because I probably have an exam today.
Mom: Yes, but you really need to be careful on the way. I really don't want to see you slipping. You'd better put on your rain boots.
Son: The rain boots are not in good quality. I don't want my feet wet.
Mom: How about taking a taxi?
Son: I don't want it to be expensive to get to school.
Mom: Ok, let your dad drive you there!
"
It is a common dialogue of a mom and her son discussing the way to get to school. We cannot exactly follow the scale but we know by intuition that the rank of necessity (here, not the absolute value) may be: "probably" > "don't" > "really don't". And in a single dialogue, only the rank or the relative value of necessities matter. So, we just need to set the necessities of situations following this rank with one more constraint: for the negative/positive probability words the necessities should be negative/positive.

And we consider the second dialogue in the following implementation.

Implementation (Qingjie)

We set a scale for 7 different values of necessities for 7 representative probability words.

Probability words	Necessities
certainly	30
highly_likely	20
probably	10
even	0
probably_not	-10
unlikely	-20
chances_are_slight	-30

In the example dialogue, “probably” corresponds to “probably” of course, “really don’t” corresponds to “chances_are_slight”, and “don’t” corresponds to “unlikely”.

We have adapted the lab codes of argumentation. We implement a new mini-knowledge base corresponding to our example dialogue, by setting the initial facts, the actions, and the causal clauses. The only difference in the knowledge base is that we don’t set any preference as we will set it on the fly. Our program is to provide the solutions of each conflict and it will finally give “dad_drives” as the best solution to attend an exam in the heavy rain.

The first step of our program is to set the necessities of each affair. The program asks the preference for “attend_exam”, “slip”, “wet_feet”, and “expensive”. And we should answer in the 7 probability words above.

Then, the program iteratively does the CAN procedure to solve the conflicts.

At last, it gives us the final solution: “dad_drives”.

Results

```
1 ?- go.
 How is your preference for attend_exam? Enter [certainly, highly_likely, probably,
                even, probably_not, unlikely, chances_are_slight].
|: probably.
```

How is your preference for slip? Enter [certainly, highly_likely, probably,
even, probably_not, unlikely, chances_are_slight].
|: chances_are_slight.
How is your preference for wet_feet? Enter [certainly, highly_likely, probably,
even, probably_not, unlikely, chances_are_slight].
|: unlikely.
How is your preference for expensive? Enter [certainly, highly_likely, probably,
even, probably_not, unlikely, chances_are_slight].
|: unlikely.
(Re)start...
Conflict of intensity 10 with attend_exam
on_foot is revisable because it is an action with no prerequisite
------> Decision : on_foot
(Re)start...
Conflict of intensity -30 with slip
Negating storm , considering -storm
Giving up: storm is stored with necessity 30
(Re)start...
Conflict of intensity -30 with slip
-on_foot is revisable because its status is unknown
------> Decision : -on_foot
(Re)start...
Conflict of intensity 10 with attend_exam
put_on_rain_boots is revisable because it is an action with no prerequisite
------> Decision : put_on_rain_boots
(Re)start...
Conflict of intensity -20 with wet_feet
-put_on_rain_boots is revisable because its status is unknown
------> Decision : -put_on_rain_boots
(Re)start...
Conflict of intensity 10 with attend_exam
take_taxi is revisable because it is an action with no prerequisite
------> Decision : take_taxi
(Re)start...
Conflict of intensity -20 with expensive
-take_taxi is revisable because its status is unknown
------> Decision : -take_taxi
(Re)start...
Conflict of intensity 10 with attend_exam
dad_drives is revisable because it is an action with no prerequisite
------> Decision : dad_drives
(Re)start...

true.

Discussion (Qingjie and Temur)

What we achieve and what we don't

Our model achieved, to some extent, the setting of necessity on the fly based on a real dialogue. And the further CAN procedure works well to give solutions and it is also compatible with a solution given by humans.

However, the setting of necessity is still partially artificial. The ideal program should for example automatically extract the necessity of "slip" as -30 from the sentence "I really don't want to see you slipping."

Future work

We may try parsing methods to analyze syntactically the structure of a sentence and to identify the probability words. However, that is challenging because how we parse the sentence is defined by the grammar, but the probability words are not of the same part of speech ("probably" is an adverb but "don't" is an auxiliary verb)!

To solve the problem, we may just identify the probability words by traversing the sentence. However, not all the probability words indicate necessity ("don't" in "I don't know" doesn't indicate necessity)!

Actually, at the beginning of this project, we wanted to implement the previous ideas, but it turned out to be incomplete and insufficient. Maybe to separate different cases and to locate the sentence with affairs can be a solution.

Bibliography (Qingjie and Temur)

[1] "Belief–desire–intention software model, Wikipedia, Wikimedia Foundation, 24 May 2021, https://en.wikipedia.org/wiki/Belief%E2%80%93desire%E2%80%93intention_software_model".

[2] Fürnkranz J., Hüllermeier E. (2011) Preference Learning. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_662.

[3] W. Fagen-Ulmschneider, "Perception of Probability Words, https://waf.cs.illinois.edu/visualizations/Perception-of-Probability-Words/".