Easy contextual intent prediction and slot detection

Aditya Bhargava, Asli Celikyilmaz, Dilek Hakkani-Tür, and Ruhi Sarikaya

Summary

We investigate the incorporation of context into the spoken language understanding (SLU) sub-tasks of intent prediction and slot detection. Using a corpus that contains information about whole sessions rather than just single utterances, we experiment with the incorporation of information from previous intra-session utterances into the SLU tasks on a given utterance.

For slot detection, we find no significant increase using CRF features indicating slots in previous utterances. For intent prediction, we achieve error rate reductions of upto 8.7% by incorporating the intent of the previous utterance as an SVM feature, and similar gains when treating intent prediction as a sequential tagging problem with SVM-HMMs.

The problems

U	**Intent**	**Transcribed**	**Recognized**
u_1	get clip	`show me the [firefly]`$_{content-name}$ `[trailer]`$_{type}$	`show me the [firefly]`$_{content-name}$ `[trailer]`$_{type}$
u_2	find info	`who directed [it]`$_{content-name-ref}$	`who directed [it]`$_{content-name-ref}$
u_3	find content	`what else has [he]`$_{director-ref}$ `done`	`what else has [he]`$_{director-ref}$ `done`
u_4	play content	`play [the avengers]`$_{content-name}$	`plane [avatars]`$_{content-name}$

Traditionally, both intents and [**slots**] are predicted per-utterance, while ignoring previous utterances within the session. However, the data is gathered not one utterance at a time but one *session* at a time; each utterance occurs in the context of a larger discourse.

We examine the effect of incorporating information from previous intra-session utterances (*ab hinc*, **context**). Context can serve as an additional source of information and help get around other errors such as those introduced during the ASR process.

Intents

- Global property of utterance
- Signify goal of user; vary by domain
- Something like determining which function to call (e.g. `find_content()`, `play_content()`, etc.)
- Traditionally an utterance classification problem

Slots

- Exist within utterances
- Local properties; slots span individual words
- Tend to be semantically loaded
- Represent actionable content, like arguments to a function (e.g. `director='Joss Whedon'` passed to a function like `find_content()`)

Session modeling

Dialog modeling also considers context (e.g. POMDPs, DBNs, etc.). We focus on incorporating context at the SLU level:

- Minimizing SLU errors prevents cascaded errors throughout the rest of the system
- Dialog modeling is not always used or needed, but context could still be helpful
- Other downstream applications that need only SLU can benefit from improved performance
- Other dialog system components can be tied to specific application scenarios or knowledge bases

Session data

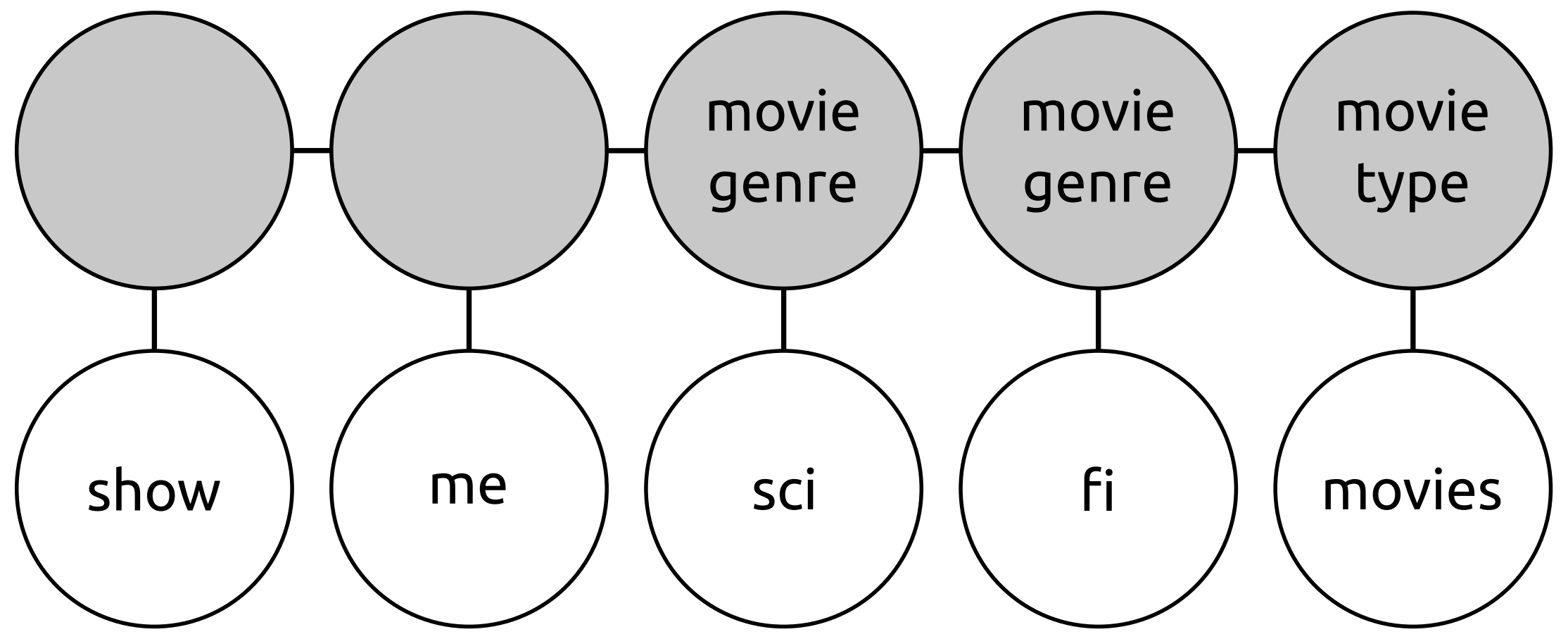
We use an internal data set collected from real user sessions. Users interact by voice with an automated system to interact with multimedia libraries.

We have 6,390 sessions with a total of 27,565 utterances. The data have 28 possible intents (*find content*, *play content*, *find similar*, *filter*, etc.) and 26 possible slot types (*content name*, *content type*, *genre*, etc.). Each utterance includes annotated intents and slots, as well as both transcribed (TRA) and speech-recognized (ASR) versions of the utterance. The session-level information indicates which utterances occur in the same session and the order in which they appear.

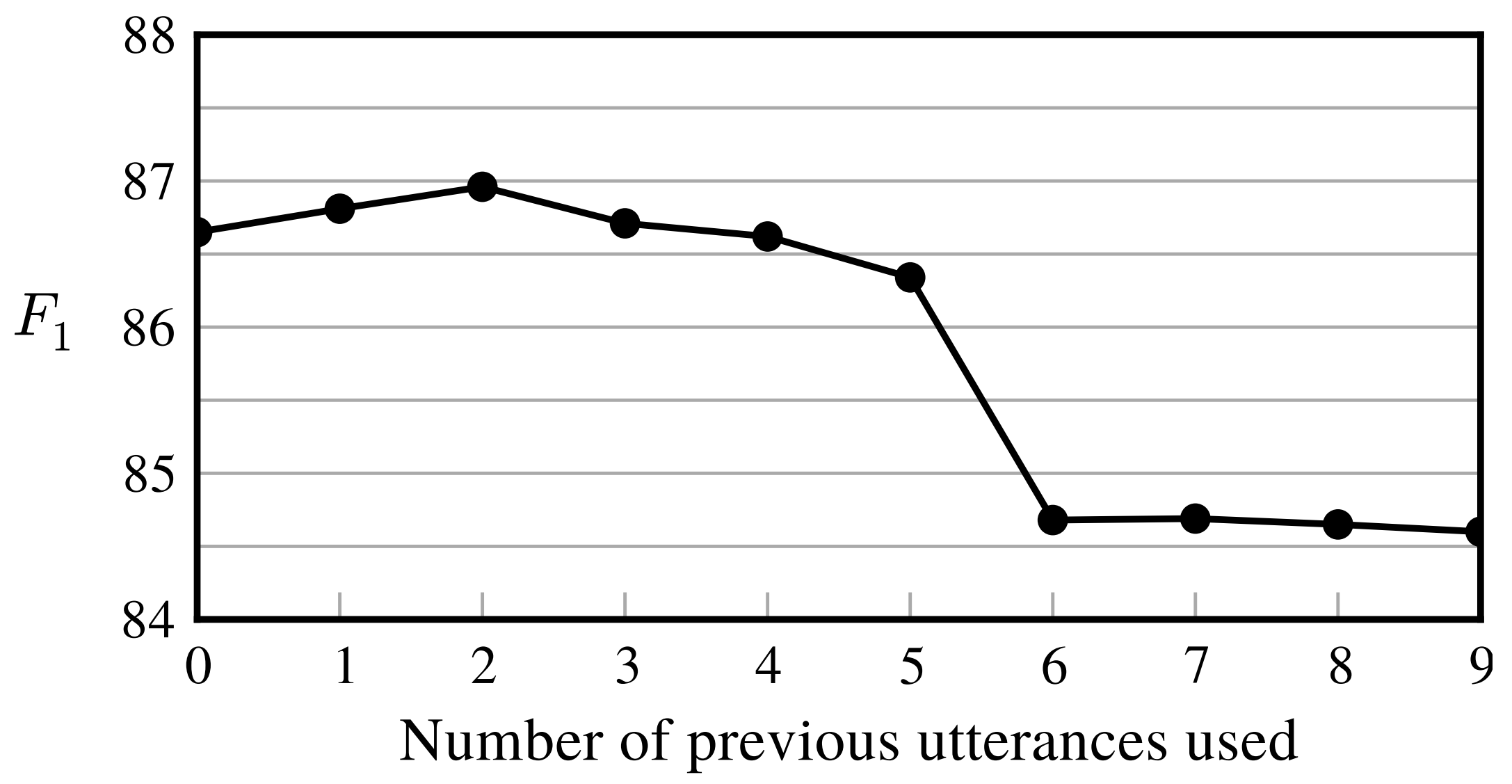
We split the data into training, development, and test sets of 80%, 10%, and 10% respectively. The development set is used for tuning hyperparameters and the test set is held out; for final testing, the development set is merged into the training set.

Slot detection

We want to incorporate information from slots found in previous intra-session utterances. Treating slot detection as a sequential tagging problem, we apply conditional random fields (hidden states are shown shaded):



Our non-contextual baseline uses only lexical features consisting of unigrams in a five-word window around the current word. Contextual information adds features for all possible slot types that might have occured in the previous n utterances.



Evaluating using F_1-score, we find a statistically insignificant increase when looking at the past two utterances, and performance decreases when looking further back than that.

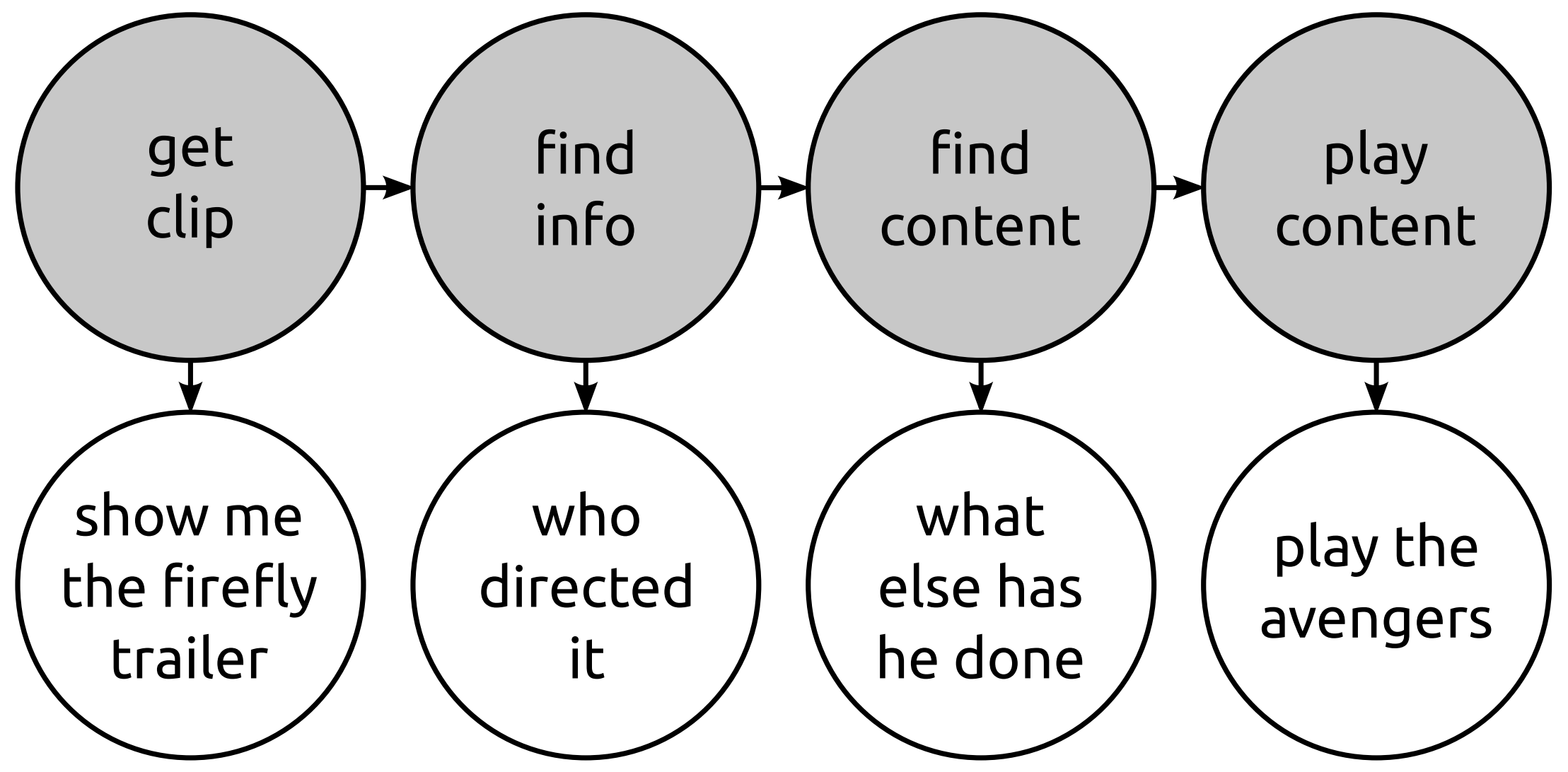
Intent prediction

We can treat intent prediction as a multi-classification problem and apply support vector machines. This allows us to easily add contextual information using a feature to represent the intent of the previous utterance. We can use both the actual intent of the previous utterance (ORCLPREV) to get a rough upper bound as well as the predicted intent (PREDPREV) for a more realistic scenario.

	TRA	ASR
BASE	97.1	93.1
ORCLPREV	97.3	93.9
PREDPREV	97.3	93.7

We find 6.7% error rate reduction in accuracy for TRA and 8.7% for ASR. PREDPREV is very close to ORCLPREV, demonstrating the efficacy of this approach.

Lastly, we treat intent prediction in a session as a sequential tagging problem with SVM-HMMs (hidden states are shown shaded):



We find no significant improvement using SVM-HMMs over our standard SVM approach.