Semantic Parsing Using Content and Context: A Case Study from Requirements Elicitation

Reut Tsarfaty, Weizmann Institute, Rehovot, Israel
Ilia Pogrebezky, Interdisciplinary Center, Herzliya, Israel
Guy Weiss, Weizmann Institute, Rehovot, Israel
Yaarit Natan, Weizmann Institute, Rehovot, Israel
Smadar Szekely, Weizmann Institute, Rehovot, Israel
David Harel, Weizmann Institute, Rehovot, Israel

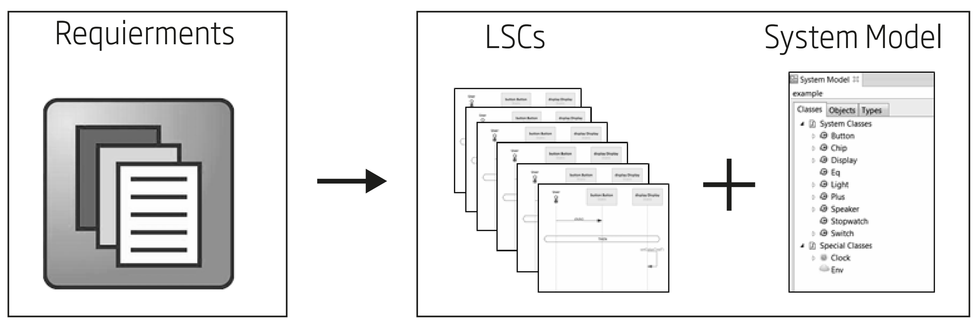
iscol
September 2014

The Challenge

The Task

The Task: Text-to-code generation

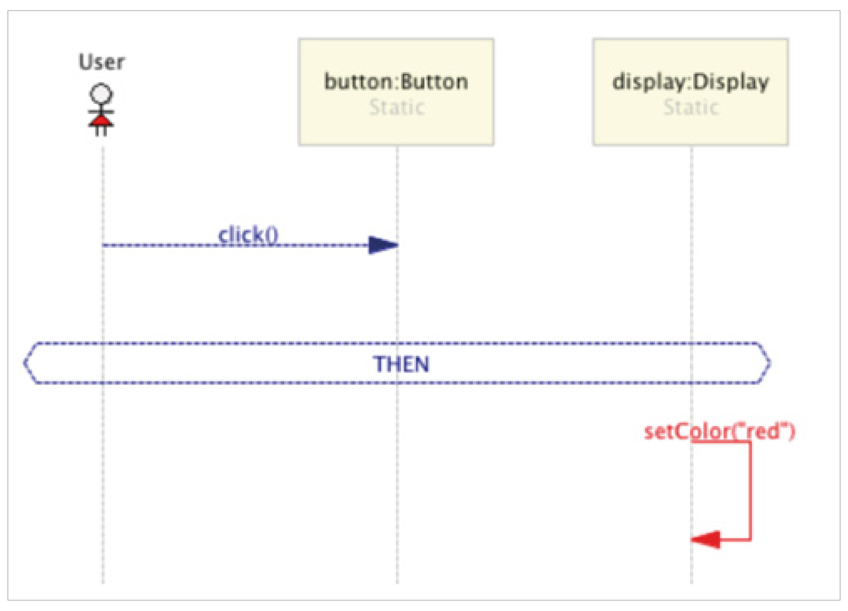
Input : A Requirements Document
Output: A working system



Target Representation

Live Sequence Charts [LSC]

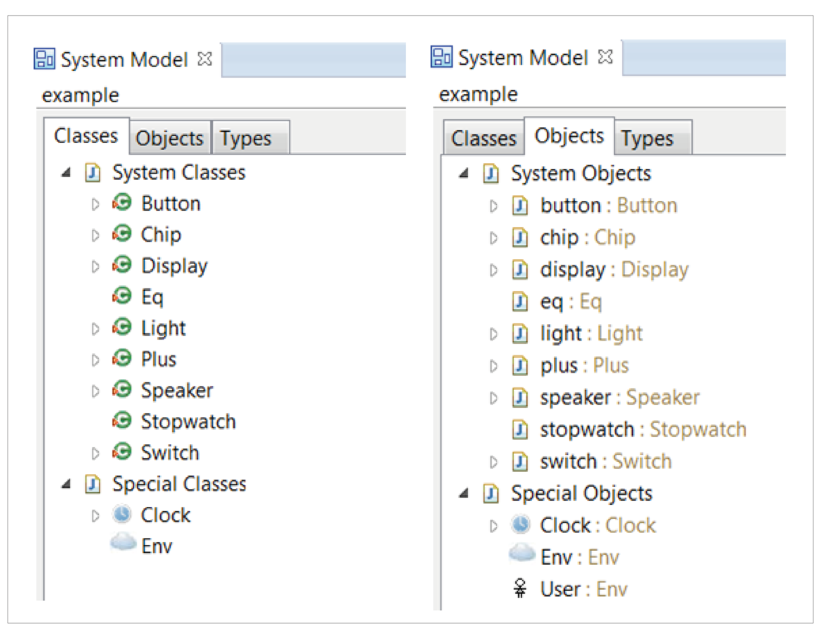
A live sequence chart (LSC) describes a possible or necessary run of a specified system. Time in LSCs proceeds from top to bottom. An LSC consists of **lifelines**, **methods**, **modalities**, and **execution status**. LSCs have a direct translation into executable code.



An LSC scenario for the sentence: "When the user clicks the button, the display color must change to red."

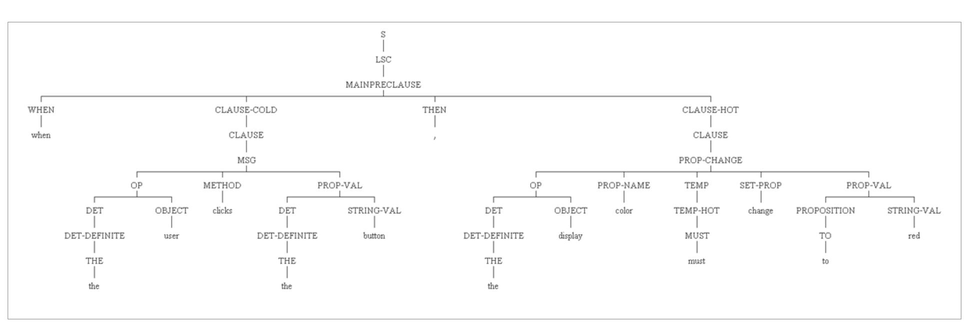
System Model (SM)

A system model (SM) presents the implemented architecture. It consists of **objects**, **methods**, **properties** and **values**.



Syntax Tree

A syntax tree (ST) is a tree representation of the abstract structure of the requirements according to a context-free grammar (Specifically, the context free grammar specified in [5])



References

[1] e. black, j. d. lafferty, and s. roukos. 1992. development and evaluation of a broad-coverage probabilistic grammar of english-language computer manuals. in proceedings of acl , pages 185–192.

[2] damm and d. harel. 2001. lscs: breathing life into message sequence charts. form. methods syst.des. , 19(1):45–80, july.

[3] m. gordon and d. harel. 2009. generating executable scenarios from natural language. in proceedings of the 10th international conference on computational linguistics and intelligent text processing , cicling '09, pages 456–467, berlin, heidelberg. springer- verlag.

[4] h. p. grice. 1975. logic and conversation. in p. cole and j. l. morgan, editors, syntax and semantics:
vol. 3: speech acts , pages 41–58. academic press, san diego, ca.

[5] d. harel. 2001. from play-in scenarios to code: an achievable dream. computer , 34(1):53–60, january.

[6] h. kugler, d. harel, a. pnueli, y. lu, and y. bontemps. 2005. temporal logic for scenario-based specifications. in proceedings of the 11th international conference on tools and algorithms for the construction and analysis of systems , tacas'05, pages 445–460, berlin, heidelberg. springer-verlag.

[7] p. liang and c. potts. 2014. bringing machine learning and compositional semantics together. annual reviews of linguistics (submitted).

[8] t. parsons. 1990. events in the semantics of english: a study in subatomic semantics . mit press, cambridge, ma.

[9] c. shannon. 1948. a mathematical theory of communication. bell system technical journal , 27:379–423.

[10] r. tsarfaty, j. nivre, and e. andersson. 2012. cross framework evaluation for statistical parsing. in w. daelemans, m. lapata, and l. m`arquez, editors, proceedings of eacl , pages 44–54. the association for computer linguistics.

[11] a. viterbi. 1967. error bounds for convolutional codes and an asymptotically optimum decoding algorithm. ieee trans. inf. theor.

[12] d. h. younger. 1967. recognition and parsing of context-free languages in time n3. information and control , 10(2):189–208.

The Model

The Idea: Use Discourse Context For Disambiguating Individual Requirements

Probabilistic Modeling

"The Probabilistic Model: $f(D) = argmax_{M \in \mathcal{M}} P(M|D)$

$D \in \mathcal{D}$ is a piece of discourse consisting of an ordered set of requirements.

$M \in \mathcal{M}$ is a code base hierarchy that grounds the semantic interpretation of the requirements.

Noisy Channel Model

The objective function:
$$f(D) = argmax_{M \in \mathcal{M}} P(M|D) = argmax_{M \in \mathcal{M}} \frac{P(D|M)P(M)}{P(D)} = argmax_{M \in \mathcal{M}} P(D|M)P(M)$$

Modeling assumption 1:
The source: $P(M) = P(m_1, m_2, \ldots, m_n) \approx \prod_i P(m_i | m_{i-1}, \ldots, m_{i-k})$

Modeling assumption 2:
The channel: $P(D|M) = P(d_1, d_2, \ldots, d_n | m_1, m_2, \ldots, m_n) \approx \prod_i P(d_i|m_i)$

Hidden Markov Model

Assuming a bigram, model our objective function is as follows:

$$f(D) = argmax_{M \in \mathcal{M}} \prod_i \underbrace{P(m_i|m_{i-1})}_{\text{transitions}} \underbrace{P(d_i|m_i)}_{\text{emissions}}$$

Our model is a hidden markov model (HMM)
- transition probabilities: represent the gaps between SM snapshots of adjacent requirements.
- emission probabilities: model the verbal description of each requirement.

The Solution

Parameter Estimation

Emission probabilities: $\hat{P}(d|m) = \frac{\hat{P}(d,m)}{\hat{P}(m)} = \frac{\sum_{t \in \{t | yield(t)=d, m \triangleright sem(t)\}} \hat{P}(t)}{\sum_{t \in \{t | t \in T, m \triangleright sem(t)\}} \hat{P}(t)}$

P(t) is the probability of a syntax tree.
m ⊳ sem(t) means that the LSC semantics of the syntax tree is grounded in the system model m [see formal details in the paper].

We sum the probability of all the trees that derive the requirement d and have LSC semantics that is grounded in the system model m, and normalise it by the sum of all possible trees that their LSC semantics is grounded in m.

We use maximum likelihood estimation (MLE) to learn the tree probabilities P(t):
> Input: a set of syntactically annotated trees
> Output: a probabilistic context free grammar
We allow for an open-ended lexicon and learn a distribution for unknown-words smoothing.

Transition probabilities: $\hat{P}(m_i|m_j) = \frac{gap(m_i|m_j)}{\sum_{m_j} gap(m_i|m_j)}$

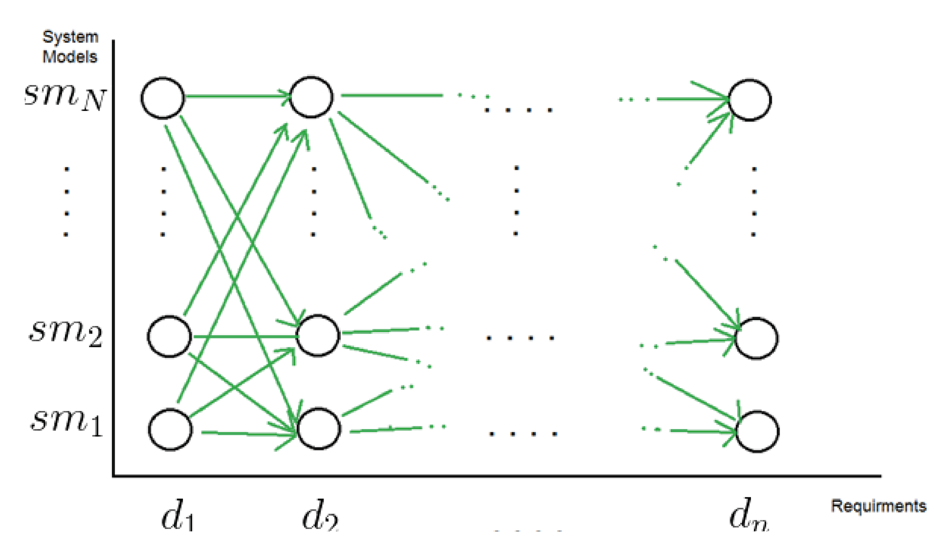
where gap(.) quantifies the information sharing between SM Snapshots.

Transition:	$gap(m_{curr}, m_{prev})$
max-overlap	$\frac{\lvert set(m_{curr}) \cap set(m_{prev}) \rvert}{\lvert set(m_{curr}) \rvert}$
max-expansion	$1 - \frac{\lvert set(m_{curr}) \cap set(m_{prev}) \rvert}{\lvert set(m_{prev}) \cup set(m_{curr}) \rvert}$
min-distance	$1 - \frac{ted(m_{prev}, m_{curr})}{\lvert set(m_{prev}) \rvert + \lvert set(m_{curr}) \rvert}$

The Decoding Algorithm

Recall: Our objective function is
$f(D) = argmax_{M \in \mathcal{M}} \prod_i P(m_i|m_{i-1}) P(d_i|m_i)$

Where $P(m_i|m_{i-1})$ are the transition probabilities in Viterbi (of the arrows) , and $P(d_i|m_i)$ are the emission probabilities in Viterbi (of the states).



emission probabilities
transition probabilities

Finding the most probable path of SM snapshots

$$\textbf{Viterbi}: \{(d_i, ST_i) | 1 \le i \le n\} \rightarrow: \{(d_i, st_i) | 1 \le i \le n\}$$

- The algorithm chooses the best syntax tree per requirement by using the context of previous requirements.
- The context is given by the system model of each possible syntax tree, Using function $g: syntax\ tree \rightarrow system\ model$
- Complexity – poly(n,N).

Finding the k-Best syntactic trees for each requirement

$$\textbf{CKY}: (\{d_1, \ldots, d_n\}, N) \rightarrow \{(d_i, ST_i) | 1 \le i \le n\}$$

- ST_i is a non-empty set set set of syntax trees that returned by CKY, $|ST_i| \le N$. (The algorithm returns the N-best syntax trees per each requirement d_i)
- Complexity – poly(n, |G|, N, l) where |G| is the size of the grammar, and l is the maximum requirement length over all requirements.

Book	the	flight	through	Houston
S, VP, Verb Nominal, Noun [0,1]	[0,2]	S,VP,X2 [0,3]	[0,4]	S,VP,X2 [0,5]
	Det [1,2]	NP [1,3]	[1,4]	NP [1,5]
		Nominal, Noun [2,3]	[2,4]	Nominal [2,5]
			Prep [3,4]	PP [3,5]
				NP, Proper-Noun [4,5]

Empirical Evaluation

Experimental Setup

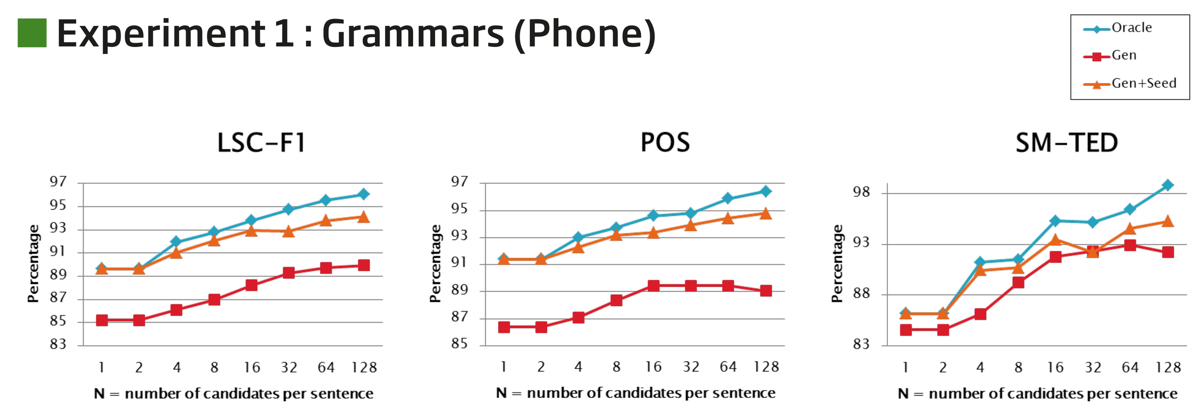
> Data
- synthetic (automatically generated) example (10000 requirements)
- 4 hand-annotated case studies (~100 requirements)
  * Phone (development set)
  * Wrist Watch
  * Chess
  * Vending Machine
  * Baby Monitor

> Grammar Estimation
- generated only
- generated + seed

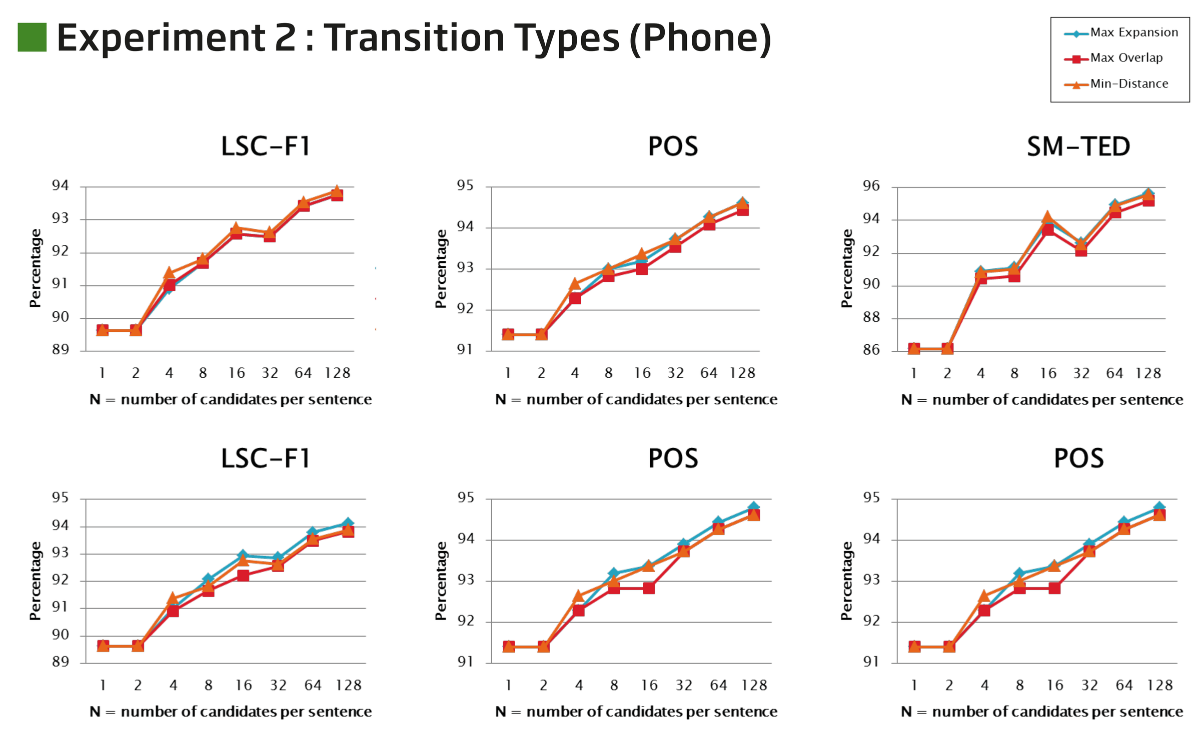
> Transition Estimators
- Max Overlap
- Max Expansion
- Min Distance
- Hybrid mode

> Evaluation Metrics
- **POS** : the accuracy of correctly predicted part-of-speech tags
- **LSC-F1** : ParsEval [1] on the predicted LSC syntax trees
- **SM-TED**: Tree-edit distance on the predicted SM trees

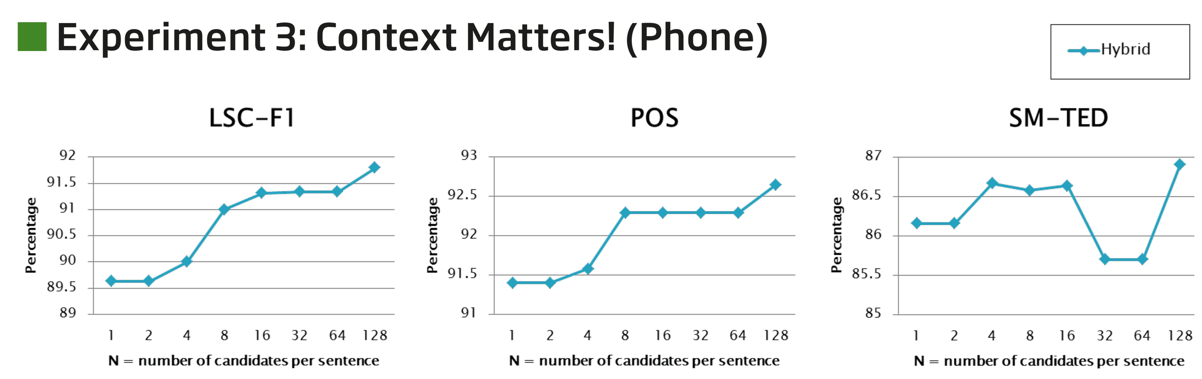
Experiment 1 : Grammars (Phone)



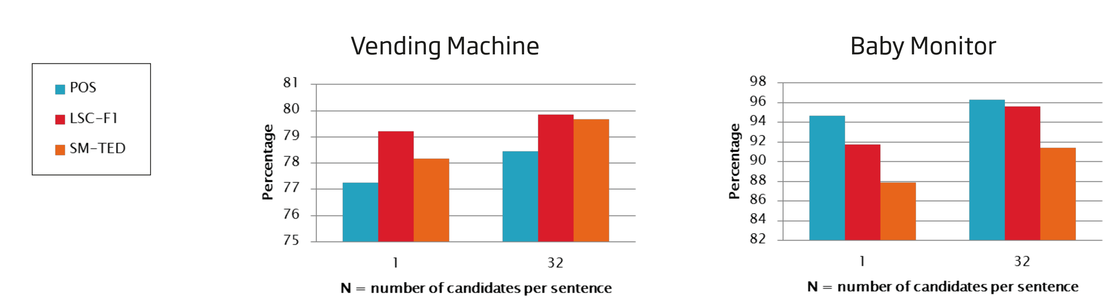
Experiment 2 : Transition Types (Phone)

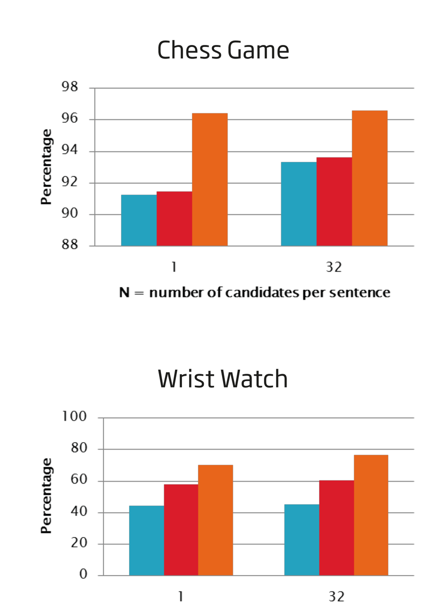


Experiment 3: Context Matters! (Phone)



Experiment 4 : Cross-Fold Validation





Future Work

> Language:
CNL ⟶ NL
> Modeling:
HMM ⟶ CRF
> Learning:
Generative ⟶ Discriminative
> Data:
Case studies ⟶ Real Application