A Comparative Test Between Industry Leading NLP Technologies And Their Intent Engines

THE CHALLENGERS:

Cognitive Code SILVIA, Microsoft Luis,
Amazon Alexa Skills Set, and IBM Watson Assistant

Silvia
CONVERSATIONAL VOICE TECHNOLOGY

IBM Watson

Microsoft

amazon

A CONVERSATION WITH DEBORAH DAHL, Ph.D. CONVERSATIONAL TECHNOLOGIES.

We went to Deborah with a request to hire her to compare and contrast Cognitive Code's "SILVIA" Intent engine with those of all the major conversational technologies' Intent capabilities. The reason an Intent Engine is important in the world of NLP, is because without getting the users' intent correct, there's no meaningful conversation.

COGNITIVE CODE
Deborah, we can't thank you enough for taking on this assignment. What engines are we using for this comparison?

DEBORAH DAHL
Cognitive Code SILVIA
Microsoft Luis
Amazon Alexa Skills Kit
IBM Watson Assistant

COGNITIVE CODE
So, what kind of test did you set up?

DEBORAH DAHL
Each assistant was provided with a standard dataset consisting of 206 training utterances in a pizza ordering application. Each assistant was trained to process these utterances using its own training tools. Once the assistants were trained, they were tested on an additional 46 previously unseen utterances in the same domain. The assistants were scored on how accurately they found the intents and slots for the test utterances. Since the training data was the same for each assistant, differences in the scores reflect differences in the systems' underlying Natural Language Processing abilities and training efficiency.

A pizza ordering dataset was developed for the test. The dataset included the following seven intents;

AddToppingToMenu: to be used by a manager to add a new topping to the menu
Example: i'd like to add artichoke hearts and meatballs to the menu. This is Brian please

AddToppingToPizza: add additional toppings to an existing order
Example: I would like to add green peppers anchovies and pepperoni please

OrderPizza: state a pizza order
Example: hi i want to have a large pizza with double cheese tomato sauce and regular crust for delivery please

SpecialsQuery: ask about sales and specials
Example: do you have any specials today

StartOver: restart the order
Example: let's start over

TellJoke: tell a joke
Example: can you tell me a joke

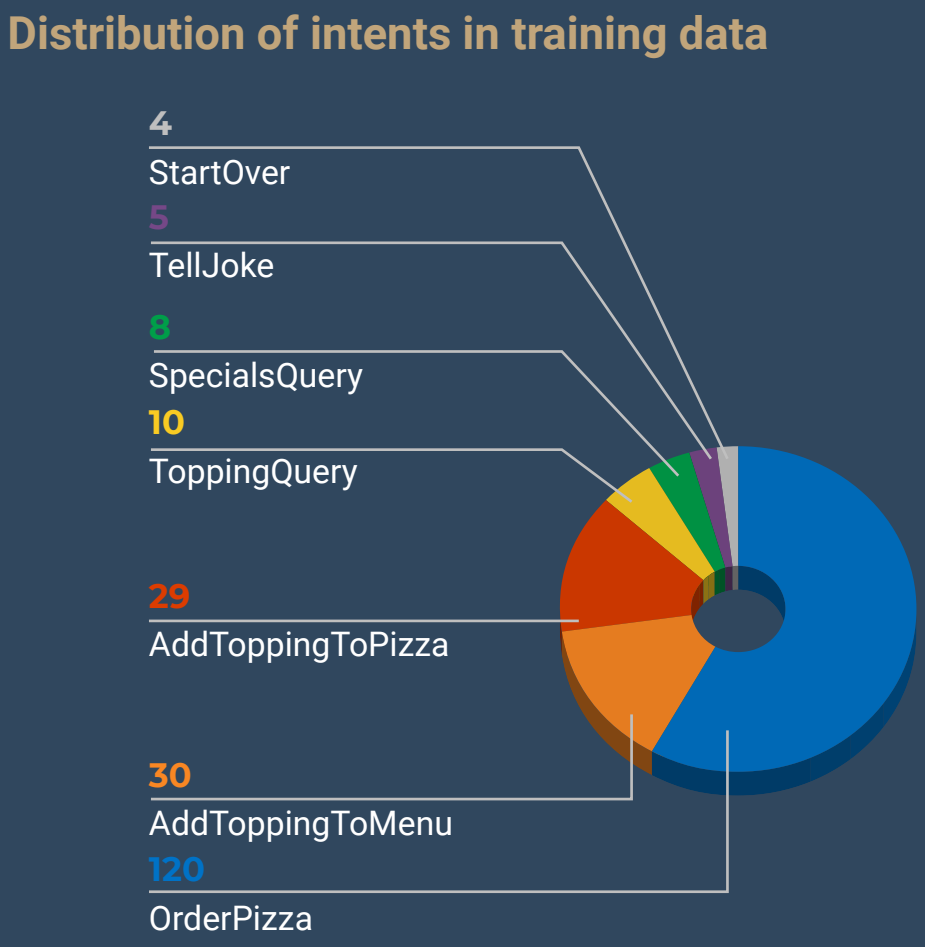
ToppingQuery: ask about the availability of toppings
Example: hi is pepperoni available please

COGNITIVE CODE

That seems pretty scientific. So, did you ask the same number of questions of each of the seven intents?

DEBORAH DAHL

As is normally the case for real world data, the intent categories are not evenly distributed. Here's a graph showing the distribution.

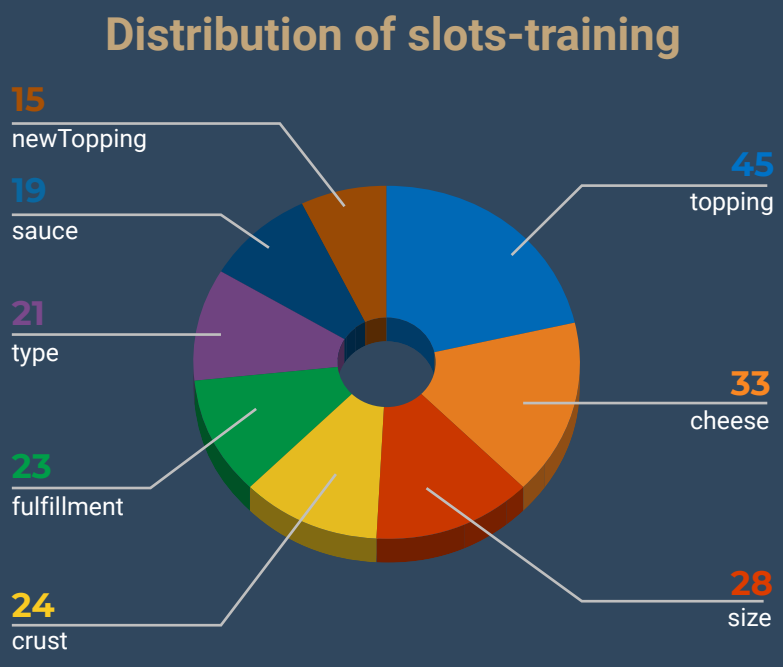


With machine-learned applications, this typically results in lower accuracy for the intents with fewer training examples. This is the case in the current evaluation.

In addition to the intents, supplementary information is supplied in the form of slots (or entities) and their values. These are the 8 entities and values that occur in the data. Some of the values have synonyms (for example, “BBQ/barbecue”). These are considered to be equivalent.

Entities	Values									
topping	sausage	pepperoni	ham	pineapple	mushrooms	onions	Green peppers	anchovies		
crust	regular	thin	thick	Chicago style						
fulfilment	take out	delivery								
size	small	medium	large	extra large						
cheese	extra	no	double							
sauce	bbq	white	tomato	regular						
type	veggie	hawaiian	margherita	four cheese						
new Toppong	bacon	ground beef	shrimp	artichoke hearts	arugula	grilled chicken	meatballs	black olives	Roasted garlic	jalapenos

The slots are fairly evenly distributed in the training data, ranging from 15 occurrences of “newTopping” to 45 occurrences of “topping”.



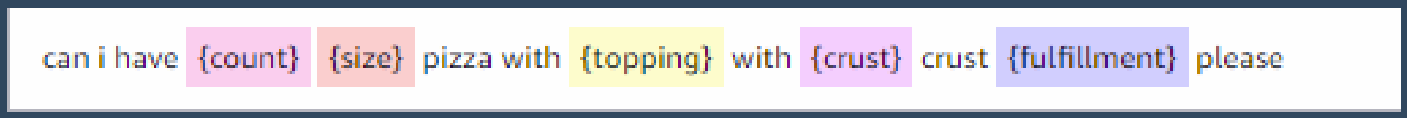
COGNITIVE CODE

So how did you evaluate the four engines. Did you have to invent something special for this?

DEBORAH DAHL

We followed the standard training/testing process for natural language understanding evaluations. Systems are provided with a set of representative examples of utterances that would be used in the application, which the system uses to develop a model. To test the system, the trained application is presented with new utterances, the test set, that did not occur in the training set, but which are representative of the application utterances. Standard practice is for the test set to be much smaller than the training set.

For Amazon, an Alexa Skills Kit application was developed using the training data and the Alexa Skills kit instructions as described at ***https://developer.amazon.com/en-US/alexa/alexa-skills-kit*** .
The developer annotates each utterance in the training data set with the correct intent and slots using the Alexa GUI interface, for example:



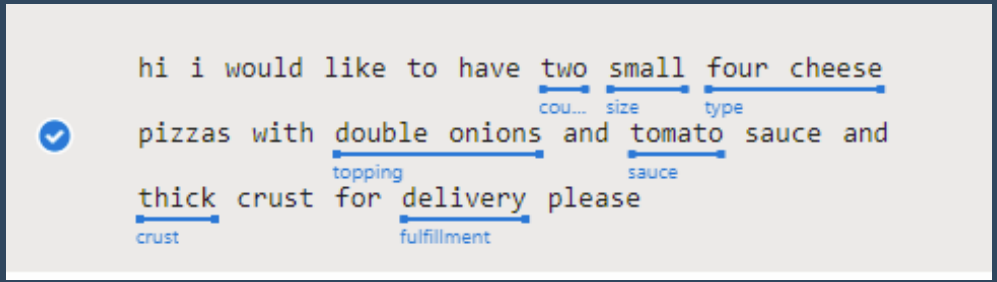
Note that the Alexa developer's GUI interface shows only the slot names, not the values.

Alternatively, Alexa provides a JSON format for batch uploads of formatted training data.

A machine learning algorithm uses the annotated data to develop a natural language model that can recognize new instances of the intents and slots at runtime.

For Microsoft, A Microsoft LUIS application was developed using the training data and the LUIS instructions at https://www.luis.ai.

Like the Alexa development process, the developer annotates each utterance in the training data set with the correct intent and slots using the LUIS GUI interface, for example:



Alternatively, LUIS provides a JSON format for batch uploads of formatted training data.

A machine learning algorithm uses the annotated data to develop a natural language model that can recognize new instances of the intents and slots at runtime.

For Watson Assistant, the offering is very easy to learn but is quite short on flexibility, so I had to forgo some validations, but filing out intents and entities was simple.

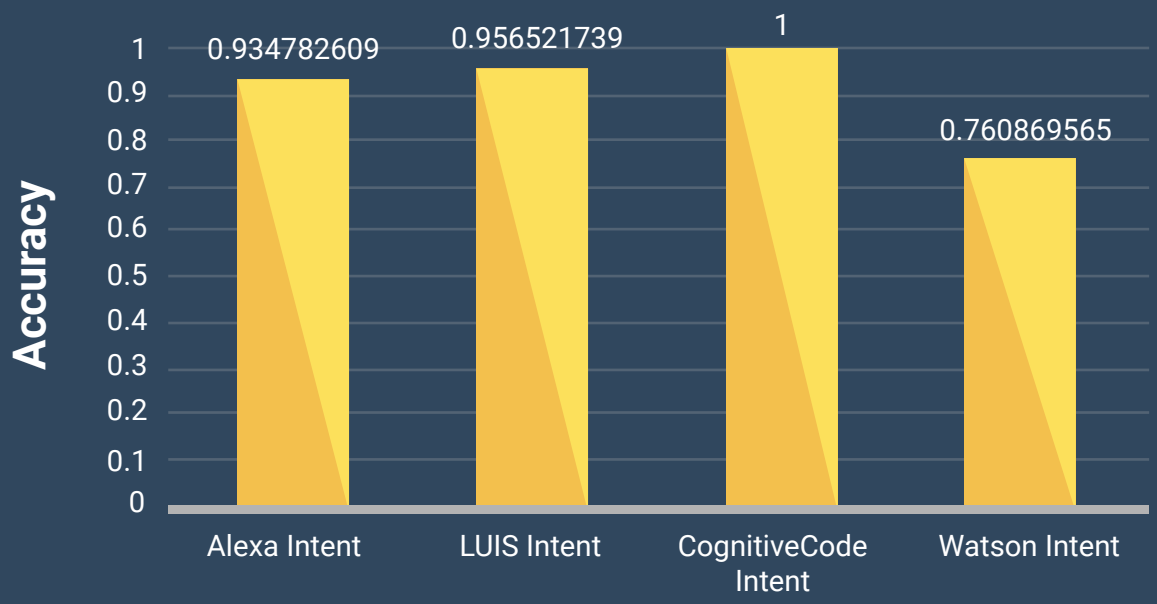
For Cognitive Code's SILVIA, we wanted to maximize Deborah's time to review and testing, so we hired a consultant with experience in developing conversational applications with Cognitive Code SILVIA and IBM Watson to build those applications, to work alongside Deborah on this project. According to his response:

"For Cognitive Code SILVIA, the application was fairly simple using just the unscripted behaviors (intents) and Entities, where I could set up a "Toppings" Entity and then assign all of the offered toppings with a binding of "isA", so validating and suggesting toppings was easy. C# scripts, available for every behavior, kept track of which slots had been filled and which slots were mandatory vs non-mandatory".

COGNITIVE CODE
OK, so how were the actual tests performed?

DEBORAH DAHL
The test set was supplied to each trained system through its testing tools, and the results were tabulated. The actual intents were compared to the expected intents. If they differed, the system was counted as having made an error on that utterance.

Similarly, the expected and actual slots were compared, as well as the slot values. For slots, it was counted as an error if the slot was missing, or if the slot name was incorrect. For values, it was counted as an error if the slot value was missing or incorrect. Here are the results.



System

Accuracy is based on the 46 occurence of intents in the test utterances

COGNITIVE CODE

Wow, so Cognitive Code beat Alexa, Luis, and Watson on Intent accuracy?

DEBORAH DAHL

For the Intent accuracy test, yes, Cognitive Code's SILVIA scored the highest out of the four tested with zero errors. The Amazon application made 3 errors, and the Microsoft application made two errors. The high error rate of the Watson application (14 errors) was primarily due to its inability to dynamically add new vocabulary, which is required by the "AddToppingToMenu" intent.

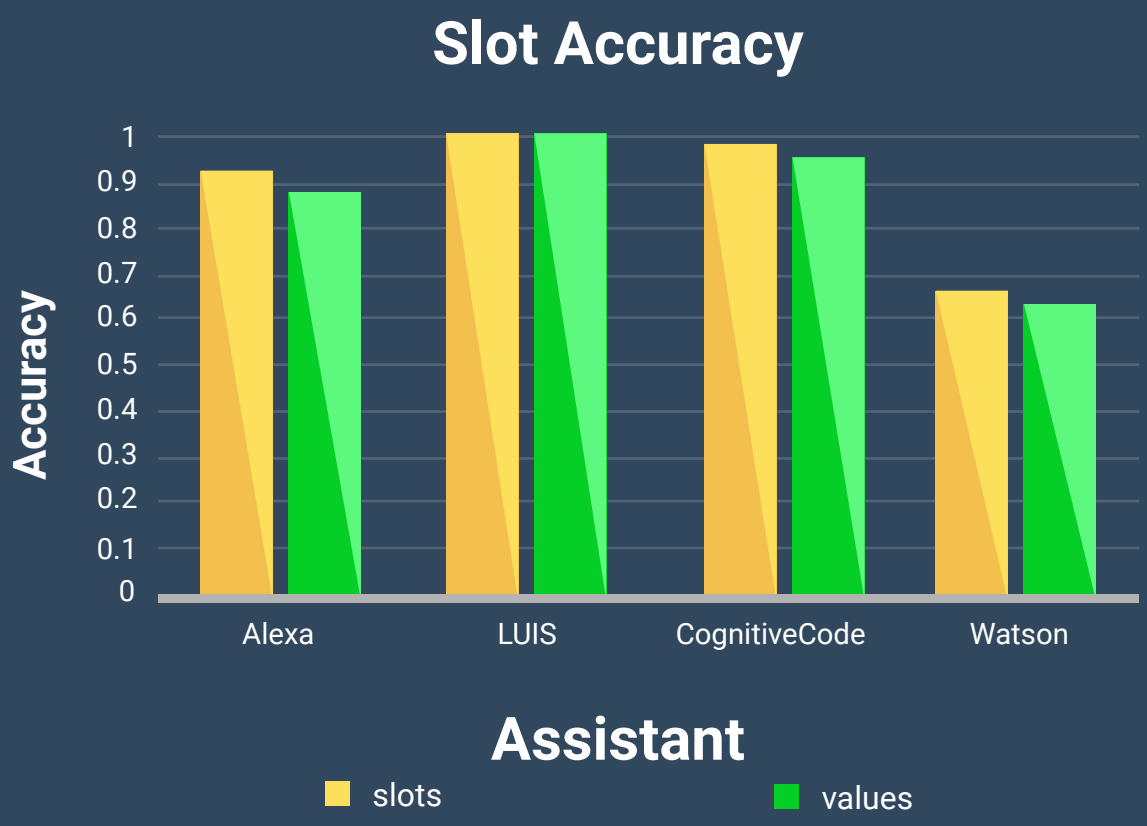
The test set included a total of 181 slots and values. A slot error occurred when the slot did not occur in the result, although it did occur in the utterance. For example, in the test utterance, "hi i need to get some extra-large veggie pizzas with double anchovies with regular sauce thin crust and no cheese I'm going to pick that up please", the Alexa application failed to find the "fulfillment" slot.

A slot value error occurred either when a slot had no value or the wrong value. For example, in the test utterance, "hi i want to have two extra-large margherita pizzas with extra onions and Chicago style crust and regular sauce" the Alexa application filled the "crust" slot with the incorrect value, "Chicago style regular".

FOR SLOTS, the best performance in the slot accuracy was by Microsoft LUIS, which got 98% of the slots correct. Cognitive Code's SILVIA came in right behind Microsoft in the slot accuracy test. Alexa made many errors because it was not

able to separate a list of several toppings into individual toppings. It also had problems recognizing the values of the “sauce” slot, which seem to have been due to the fact that “regular” was a possible value of both “sauce” and “crust”.

Here is the breakdown of accuracy of slots and values by system.



COGNITIVE CODE

We’ve seen this before with other systems. Compound entities (two or more words) are hard for some systems to handle, and some force you to use hyphens or underscores. Our ability to handle compound entities, entity relationships beyond synonyms, and our c# scripting really set Silvia apart in providing a full, robust toolkit.

DEBORAY DAHL

Overall, all of the systems performed reasonably well, with Cognitive Code’s SILVIA intent engine being the most successful. Most of the other competiotrs errors seem to have been due to missing basic capabilities rather than natural language processing errors. For example, Watson was not able to process any “newTopping” utterances because it cannot add dynamic vocabulary. Similarly, Alexa could not handle lists of several values of “topping”. The very low frequency intents were also problematic for all systems, which is to be expected with machine-learned applications, since they perform better with more training data. Similarly, the Alexa application made a mistake with “can you tell me what

the toppings are", thinking that the intent was "TellJoke", which has only 5 examples in the training data. It was likely confused by the joke training utterance, "can you tell me a joke".

There are also examples of pure natural language processing errors that did not seem to be due to a lack of training data. An example of a natural language processing error is the LUIS processing of "can i get one extra large veggie pizza with extra anchovies with regular crust tomato sauce and extra cheese take out please", where the result simply omitted the "crust" slot.

The differences between Alexa and LUIS are very small and are not likely to be meaningful. The Watson results are demonstrably worse than the other systems. While Watson's poor performance on intents is primarily due to its inability to dynamically add vocabulary, its slot errors appear to be due to some basic problems with its algorithms. For example, it often inserted a spurious "size:medium" slot:value pair, even when an actual size was mentioned in the utterance. It also frequently failed to identify slots that were mentioned.

In summary, overall Cognitive Code's SILVIA intent engine stood out and performed better than the other three. Alexa and LUIS performed relatively well, with scores in the high 80's to high 90's. Watson was quite a bit worse. Not only did it fail on the "AddNewToppingToMenu" intent, but it made many more errors in slot-filling.

COGNITIVE CODE

Thanks so much, Deborah. Excellent work. When you look past the intent and entity superiority of Cognitive Code's Silvia Intent Engine, and expand to features like it's PORTABILITY (runs natively on almost any OS and hardware with no internet required); SECURE (encrypted data with no internet connection required); CONVERSATIONAL (built for multiple turns); and PERSONALIZATION (learns and adapts to user preferences), many people have expressed to us that we have the most robust and full tool set in the industry for developing complex or simple conversational applications.

About Deborah Dahl: Deborah focuses on implementations of innovative, practical and scalable conversational systems that push the boundary between theory and applications. She is the Principal of Conversational Technologies, which assists its clients in creating state of the art solutions using speech, natural language, and dialog technologies. Her clients range in size from startups to large enterprises and government organizations. She has written many technical papers as well as three books, including Multimodal Interaction with W3C Standards: Toward Natural Interfaces to Everything, published in 2016. Dr. Dahl has a Ph.D. in linguistics from the University of Minnesota with post-doctoral

studies in cognitive science at the University of Pennsylvania.