A Semi-Supervised Clustering Approach for Semantic Slot Labelling

Heriberto Cuayáhuitl, Nina Dethlefs, Helen Hastie
School of Mathematical and Computer Sciences
Heriot-Watt University, Edinburgh, United Kingdom
Email: hc213@hw.ac.uk

***Abstract*—Work on training semantic slot labellers for use in Natural Language Processing applications has typically either relied on large amounts of labelled input data, or has assumed entirely unlabelled inputs. The former technique tends to be costly to apply, while the latter is often not as accurate as its supervised counterpart. Here, we present a semi-supervised learning approach that automatically labels the semantic slots in a set of training data and aims to strike a balance between the dependence on labelled data and prediction accuracy. The essence of our algorithm is to cluster clauses based on a similarity function that combines lexical and semantic information. We present experiments that compare different similarity functions for both our semi-supervised setting and a fully unsupervised baseline. While semi-supervised learning expectedly outperforms unsupervised learning, our results show that (1) this effect can be observed based on very few training data instances and that increasing the size of the training data does not lead to better performance, and (2) that lexical and semantic information contribute differently in different domains so that clustering based on both types of information offers the best generalisation.**

I. Introduction

Natural language processing modules are often trained from labelled examples and therefore rely on the existence of human-annotated corpora, potentially ignoring substantial amounts of unlabelled data. While, ideally, trained components will show some generalisation across domains, in reality the best results are still achieved based on domain-specific training data. In this paper, we therefore investigate a method for the automatic labelling of language corpora based on a minimal set of labelled examples. Our scenario is an NLP component within an interactive system that is able to automatically extend its domain during interactions with users. The assumption is that the system can recognise new slots[1] in user queries such as a user asking about *child-friendly* movies when this slot was not in the training data. The system is then able to retrieve the new information (e.g. from the web) and dynamically extend its domain ontology. NLP modules, such as a parser or natural language generator, then need to be re-trained online to deal with the new semantic slots. The focus of this paper will be to label the required training data with little human intervention.

Several authors have recognised the need to move away from methods that require extensive human annotations to train NLP modules. Instead, recent work has explored alternative techniques that require less supervision. This includes learning from trial and error [1], [2] in physically-situated scenarios, or learning from parallel corpora [3], [4] or databases [5], [6]. While trial and error learning is usually not an option in scenarios involving interaction with humans—because of the high number of training episodes needed—learning from parallel corpora or databases is restricted to domains for which such resources exist.

This paper, which is an extension of an unsupervised clustering approach described in [7], aims to find an alternative approach that does not rely on the existence of prior resources. We investigate a method based on semi-supervised clustering that works in four steps. We first train Bayesian networks from a minimal set of labelled examples. The trained Bayes nets are used to estimate an affinity metric capturing the similarity between simple sentences (clauses). In a second step, the affinity metric is used as part of a spectral clustering algorithm that finds clusters of clauses in the training data that share a semantic label. The crux of our method is to find a suitable similarity metric that allows the estimation of semantically meaningful clusters. In a third step, we train an additional Bayes net for identifying phrases within clauses that represent slot values. In the final step, the identified phrases are labelled with their corresponding cluster / semantic label. The annotations obtained in this way can serve as training data for various NLP components, such as semantic parsers and natural language generators within interactive conversational systems.

We test our proposed method in two inherently different domains—restaurants and movies. Three main results can be observed. First, a semi-supervised learning scenario performs significantly better than a fully unsupervised scenario based on as few as 10 labelled training instances. Second, increasing the size of labelled examples does not lead to a substantial increase in performance afterwards. Third, lexical and semantic information do not contribute equally to the similarity functions used for different datasets. Rather, it seems that different datasets vary in their lexical and semantic characteristics, so that a similarity function based on both types of information is needed to generalise across domains.

II. Related Work

Recent years have seen a surge of interest in unsupervised or weakly supervised methods for NLP that learn semantic concepts or linguistic expressions from unlabelled data, and move towards replacing the expensive/impractical labelled corpora required in supervised learning algorithms. Two popular approaches to do this have been to use (a) parallel corpora [3], [4] or (b) databases [5], [6] instead of annotations. For

[1] We use the term *slot* and *slot type* interchangeably for representing a set of slot values, e.g. filmgenre={action, drama, adventure, comedy, crime, ...}.

example, [8] and [9] induce semantic parsers from parallel corpora that contain pairs of semantic forms and natural language realisations. [10] train a dynamic Bayesian network from semantically aligned data (a mapping from dialogue acts to surface realisations) produced by human annotators. In essence, these learning approaches reduce the problem of automatic language induction to finding a mapping between two alternative abstract representations.

A separate direction has been to learn language through observation or trial-and-error search in situated scenarios, such as route direction generation. Methods explored here include learning from experience [11], [12], learning through observation of human behaviour [13], and learning from trial and error [1], [2]. These approaches receive their supervision from the real world and are therefore often only transferable to contexts which offer directly observable rewards.

Some authors have also explored semi-supervised learning for NLP applications. For example, [14] applies semi-supervised learning to named-entity recognition and Chinese word segmentation. The approach is closely related to ours in that words are clustered in a pre-processing step. The output is subsequently fed into a supervised learning algorithm and confirms that features derived from unlabelled data can help to improve the overall accuracy of the model. [15] apply semi-supervised clustering in an active learning scenario but focus more on the learning algorithm than the NLP application. Finally, [16] compares semi-supervised learning with supervised learning and shows how unlabelled data can often help to even improve models trained from labelled data, since NLP applications are often faced with data sparsity issues.

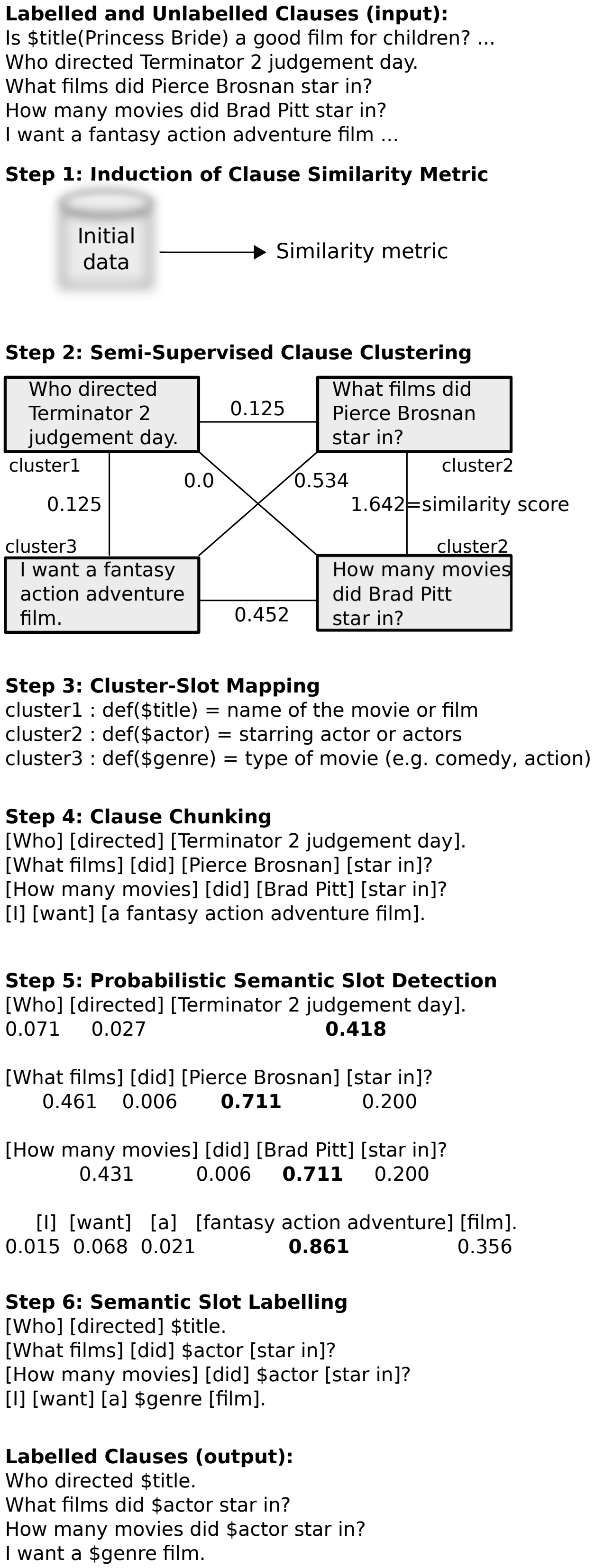
Other work related to ours includes domain adaptation. [17] presents a technique that easily transfers across domains by augmenting features from a source domain and adapting them to a target domain. Similarly, [18] address the problem of transfer from in-domain training data to out-of-domain test data using supervised learning by distinguishing general, in-domain and out-of-domain features. [19] addresses the problem of evolving intelligent systems that adjust their structure and parameters according to a changing learning environment.

Here, we present an approach to automatic semantic slot labelling based on clusters of sentences whose similarity is estimated based on lexical and semantic information that is relatively easy to obtain. The method can be seen as a means for automatic semantic annotation from raw text.

III. Method for Semantic Slot Labelling

The key idea to label semantic slots is to identify phrases in the input data that are semantically and lexically similar, cluster them, and use the clusters to map phrases onto semantic slots. Algorithm 1 shows the detailed steps involved and Figure 1 presents an illustration. This algorithm assumes that sets of labelled and unlabelled clauses are given as input, and a set of automatically labelled clauses is given as output.

- *First*, the labelled clauses (initial data) are used to induce a similarity metric that estimates the distance between unseen clauses. This is the step that makes our approach semi-supervised because we use the initially labelled data to learn a similarity metric, which is used to cluster unseen unlabelled clauses.



Labelled and Unlabelled Clauses (input):
Is $title(Princess Bride) a good film for children? ...
Who directed Terminator 2 judgement day.
What films did Pierce Brosnan star in?
How many movies did Brad Pitt star in?
I want a fantasy action adventure film ...

Step 1: Induction of Clause Similarity Metric

Initial data
Similarity metric

Step 2: Semi-Supervised Clause Clustering

Who directed Terminator 2 judgement day.
0.125
What films did Pierce Brosnan star in?
cluster1
0.0
0.534
cluster2
0.125
1.642=similarity score
cluster3
cluster2
I want a fantasy action adventure film.
0.452
How many movies did Brad Pitt star in?

Step 3: Cluster-Slot Mapping
cluster1 : def($title) = name of the movie or film
cluster2 : def($actor) = starring actor or actors
cluster3 : def($genre) = type of movie (e.g. comedy, action)

Step 4: Clause Chunking
[Who] [directed] [Terminator 2 judgement day].
[What films] [did] [Pierce Brosnan] [star in]?
[How many movies] [did] [Brad Pitt] [star in]?
[I] [want] [a fantasy action adventure film].

Step 5: Probabilistic Semantic Slot Detection

```
[Who] [directed] [Terminator 2 judgement day].
0.071   0.027                    0.418

[What films] [did] [Pierce Brosnan] [star in]?
   0.461   0.006      0.711          0.200

[How many movies] [did] [Brad Pitt] [star in]?
      0.431        0.006    0.711    0.200

   [I]  [want]   [a]  [fantasy action adventure] [film].
0.015  0.068  0.021             0.861                0.356
```

Step 6: Semantic Slot Labelling
[Who] [directed] $title.
[What films] [did] $actor [star in]?
[How many movies] [did] $actor [star in]?
[I] [want] [a] $genre [film].

Labelled Clauses (output):
Who directed $title.
What films did $actor star in?
How many movies did $actor star in?
I want a $genre film.

Fig. 1. Example clauses in the film domain showing the automatic labelling process of the proposed method. The input to this method is a small set of labelled clauses (simple sentences) and a large set of unlabelled ones, and the output is the latter set of clauses annotated with semantic slots. Although Algorithm 1 assumes a large number of sentences as input, this example illustrates the process based on a smaller set.

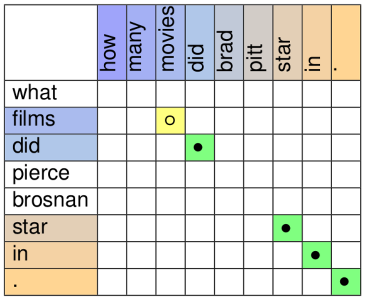
Algorithm 1 Semi-supervised labeller of semantic slots

```
1:  function SemanticSlotLabeller(List labelledClauses, List unlabelledClauses, Dictionary slots)
2:      semanticMap ← {}                                          ▷ cluster-slot mapping
3:      labelledClauses ← {}                                      ▷ output
4:      similarityMetric ← induce similarity metric from labelledClauses
5:      affinityScores ← similarity between clauses $(x_i, x_j)$, for $x_i \neq x_j$
6:      clusteredClauses ← Clustering(clauses, affinityScores, |slots|)
7:      for each cluster $g$ in clusteredClauses do
8:          $centroid$ ← average similarity of clauses in cluster $g$
9:          $s^* = \arg\max_{s_i \in slots}$ ClauseSimilarity(def($s_i$), $centroid$),
10:         where def($s_i$) is the definition of slot $s_i$ in the slots dictionary
11:         semanticMap ← APPEND($g$, $s^*$)
12:     end for
13:     for each clause $c$ in clusteredClauses do
14:         $phrases$ ← Chunking($c$)
15:         $phrase^* = \arg\max_{ph \in phrases} P(ph|$evidence$(ph))$
16:         $slotID^*$ ← semanticMap(cluster of clause $c$)
17:         $labelled$ ← clause $c$ replacing $phrase^*$ by $slotID^*$
18:         labelledClauses ← APPEND($labelled$)
19:     end for
20:     return labelledClauses
21: end function
```

- *Second*, the clauses are then grouped using unsupervised/semi-supervised clustering as described in the rest of this section. While purely unsupervised clustering will make use of a domain-independent similarity metric (see Equation 1), semi-supervised clustering will make use of the metric induced in the previous step (see Equation 6).
- *Third*, we map the clusters found onto the slot names and definitions in the system's ontology. We assume that a dictionary of slot names and their definitions is given as input, which is used to label the clusters. The lack of this dictionary would simply cause the clusters to be referred to by a non-meaningful slot name such as cluster$N$. The mapping in this step is based on the similarity between the centroids of clusters and the definitions of slots. The latter are based on phrases or keywords—see Step 3 in Figure 1.
- *Fourth*, the clauses in each cluster are then chunked into phrases using a shallow parser that uses a combination of classifiers [20]. In this step, the assumption is that some chunks will be considered slot values and the remaining chunks fillers.
- *Fifth*, all phrases found in Step 4 are classified as either representing a slot or not using a Bayesian classifier. This classifier was trained on non-lexical features of known slots in order to make it generalizable across domains, see Section III-C.
- *Last*, the slot values of all slots identified in Step 5 are replaced by their corresponding semantic slot type—as derived from the cluster-slot mapping in step 3.

A. Clause grouping using unsupervised clustering

The task of unsupervised clustering consists in partitioning clauses into maximally homogeneous groups, where homogeneity is measured based on numerical similarity. In this section, we describe a procedure for grouping clauses into k groups with equivalent semantics, corresponding to Step 3 in Algorithm 1. We assume that the number of clusters is known, and leave its automatic discovery as future work. To find a set of k clusters, we apply spectral clustering [21], though other clustering methods are possible [22].



	how	many	movies	did	brad	pitt	star	in	.
what									
films			○						
did				●					
pierce									
brosnan									
star							●		
in								●	
.									●

Fig. 2. Sample Meteor alignments, where dots are match markers. While filled dots represent exact matches, unfilled dots represent partial matches.

In spectral clustering, given a set of data points $x_1, ..., x_n$ (clauses in our case) and a pairwise affinity matrix $A_{ij} = A(x_i, x_j)$, the task is to find a set of k clusters with a clustering algorithm of preference using the data points but projected into a low-dimensional space. Such a space is obtained according to the following procedure. First, construct the pairwise affinity matrix $A_{ij} = ClauseSimilarity(x_i, x_j)$, where the affinity between clauses x_i and x_j is defined by the following cumulative scores, each score in the range [0...1]:

$$ClauseSimilarity(x_i, x_j) = MS + WRA + SSS + STS, \tag{1}$$

explained as follows. MS (Meteor Score) measures the lexical similarity between sentences (see Figure 2) calculated as

$$MS = (1 - Pen) \times F_{mean}, \tag{2}$$

where F_{mean} is a weighted precision-recall metric and Pen is a penalty that accounts for gaps and differences in word order [23]. WRA (Word Recognition Accuracy) is the complement of the well-known 'Word Error Rate' metric and also measures the lexical similarity as

$$WRA = 1 - \frac{substitutions + deletions + insertions}{|words|}. \tag{3}$$

SSS (Semafor Semantic Score) measures the semantic similarity between feature vectors (see Figure 3) $F(x_i) = \{f_0^{x_i}, ..., f_N^{x_i}\}$ and $F(x_j) = \{f_0^{x_j}, ..., f_M^{x_j}\}$ of clauses x_i and x_j, which are extracted by the Semafor Frame-Semantic Parser [24]. The score is then expressed as

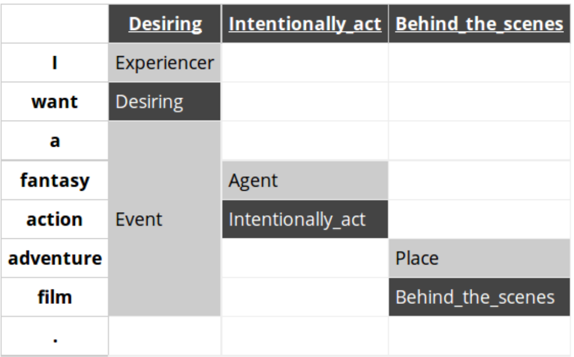
$$SSS = \frac{\sum_{m=1}^{|M|} \sum_{n=1}^{|N|} sim(f_n^{x_i}, f_m^{x_j})}{|F(x_i) \cap F(x_j)|}, \tag{4}$$

with function $sim(f_x, f_y)$ assigning 1 if semantic features $f_x = f_y$ and 0 otherwise.

Finally, STS (Semantic Textual Similarity) also measures the semantic affinity between word sequences x_i and x_j as

$$STS \approx sim_{LSA}(x_i, x_j) + 0.5 \exp^{-\alpha D(x_i, x_j)}, \tag{5}$$

where $sim_{LSA}(x_i, x_j)$ is the Latent Semantic Analysis (LSA) between clauses, $D(x_i, x_j)$ is the minimal path distance between terms within clauses derived from WordNet relations, and α is a weighting factor as described in [25]. The result of this first step is a matrix of affinity scores $A(x_i, x_j)$.



	Desiring	Intentionally_act	Behind_the_scenes
I	Experiencer		
want	Desiring		
a	Event		
fantasy		Agent	
action		Intentionally_act	
adventure			Place
film			Behind_the_scenes
.			

Fig. 3. Sample Semafor features within the clause in the first column.

As a second step in spectral clustering, we compute the Laplacian matrix $L = D - A$, where D is the diagonal matrix with element (i, i) being the sum of row i in matrix A.

Third, we compute the first k eigenvectors $V = \{v_1, ..., v_k\}$ of the Laplacian matrix L. In linear algebra, an *eigenvector* v of matrix L satisfies the property $Lv = \lambda v$, where λ is a constant called *eigenvalue*. Let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of eigenvectors V.

Last, we cluster the data points y_i into k clusters. We used the K-means algorithm with the Manhattan distance defined by $d(p, q) = \sum_{i=1}^{n} |p_i - q_i|$, where p and q are eigenvectors.

B. Clause grouping using semi-supervised clustering

We refine the unsupervised clustering approach above by using a small amount of labelled data for inducing a clause similarity metric. To do this we reformulate Equation 1 as

$$ClauseSimilarity(x_i, x_j) \approx Pr^{lex}((x_i, x_j) = \text{affine}|\mathbf{e}^{lex}) + Pr^{sem}((x_i, x_j) = \text{affine}|\mathbf{e}^{sem}), \quad (6)$$

where probabilities Pr^{lex} and Pr^{sem} are derived from querying the Bayes nets trained from the given initial labelled data, evidence $\mathbf{e}^{lex}$ refers to lexical features (in our case lemmatized words[2]), and $\mathbf{e}^{sem}$ refers to semantic features (in our case derived from the Semafor Frame-Semantic Parser [24]). A Bayes net represents a joint probability distribution based on a directed acyclic graph, where each node (i.e. a lexical or semantic feature) is associated with a probability function and connections represent dependencies. The joint probability distribution for random variables Y is defined by $P(Y) = \prod P(Y_i|pa(Y_i))$, where $pa(.)$ denotes the set of parent random variables, and every variable is associated with a conditional probability distribution $P(Y_i|pa(Y_i))$. The following tasks are involved in the creation of our Bayes net: (1) parameter learning involves the estimation of conditional discrete probability distributions from data, where we use maximum likelihood estimation with smoothing; and (2) structure learning involves constructing the dependencies of random variables based on the K2 algorithm [27]. We trained both Bayes nets using binary features and binary labels (affine or not affine). While the number of features depends on the amount of labelled examples D, the amount of training instances corresponds to $|D| \times |D|$, e.g. 10 clauses correspond to 100 instances. Once these Bayes nets have been trained, we use the junction tree algorithm [28] for probabilistic inference (probabilities of affinity between clauses). For example, given clauses x_i=*'I need help finding a soul food restaurant'* and x_j=*'Find me a fancy place to eat'*, and lexical features $\mathbf{e}^{lex}(x_i)$={food, find, i, restaurant, a, need, soul, help} and $\mathbf{e}^{lex}(x_j)$={i, find, to, eat, a, fancy, place}, results in the following inference $Pr^{lex}((x_i, x_j) = affine|\mathbf{e}^{lex}) = 0.213$. A similar inference is made on the other Bayes nets with semantic features. These inferences are used to populate the affinity matrix $A(x_i, x_j)$.

C. Semantic slot detection using supervised learning

Step 5 in Algorithm 1 identifies those phrases in the data that represent slots (in contrast to non-slot phrases). To do this, we use an additional Bayes net that was trained on features of known slots. Once this Bayes net has been trained, we use the junction tree algorithm [28] for probabilistic inference (probabilities of phrases being semantic slots within a clause). The phrase with the highest probability is selected according to $\arg\max_{ph \in Phrases} P(ph|\mathbf{e}^{ph})$, where the evidence of phrase ph is defined by $\mathbf{e}^{ph} = \{f_1 = val_1, ..., f_n = val_n\}$ with features f_i and values val_i. This Bayes net used the following features (binary except for the first two features): previous and next Part-Of-Speech (POS) tags [29], hasVerb, hasNoun, hasPronoun, hasAdverb, hasAdjective, hasPreposition, hasConjunction, phraseSize (small:$|words| \leq 4$, large:$|words|$>4), isStopWord, tf-idf level (low≤5, high>5), and label (yes, no). The last feature was used to ask probabilistic queries, e.g. 'What is the probability of this phrase being a semantic slot?', and the remaining variables were used as evidence of the phrase at hand. An example probabilistic query to the trained Bayes net to determine how likely the phrase *"Science Fiction"* is to be a semantic slot is as follows: $Pr(label = yes|\mathbf{e}(\text{“}Science\ Fiction\text{”})) = 0.679$, where evidence $\mathbf{e}$ includes feature-values such as NN=1, prevPOS=IN, nextPOS=END, phraseSize=large, hasDeterminer=no. These types of queries form the probability distribution of phrases—being semantic slots—in a clause. Classification accuracy in two domains is reported in Section IV-D.

IV. Experiments and Results

We describe an evaluation of our proposed method for automatic labelling of semantic slots, which aims to serve as evidence of its potential benefit for NLP tasks. To this end, we report: (1) the accuracy of unsupervised clustering, (2) the accuracy of semi-supervised clustering, and (3) the accuracy of the supervised classifier for slot detection.

A. The Data

Our domains are restaurants and movies, where we used the publicly available datasets from the Spoken Language Systems group at MIT[3]. Both datasets are semantically labelled in BIO format. The following is an example labelled clause with slot *Title*.

[2] We use lemmatized words (based on the Stanford NLP tools [26]) instead of raw words to represent lexical items with a more compact representation.

[3] http://groups.csail.mit.edu/sls/downloads/

```
O is
B-TITLE princess
I-TITLE bride
O a
O good
O film
O for
O children
```

The corpus used for *unsupervised/semi-supervised clustering* included 100 clauses, each clause containing 1 slot. We leave clauses containing multiple slots as future work. In addition, we used only the three most common slots in restaurants (Restaurant_Name, Amenity, Cuisine) and movies (Title, Actor, Genre).

Separately, the corpus used for *training the clause similarity metric* included different amounts of training data, i.e., 10, 20, 30, 40, and 50 clauses, to test the effect of variable training data sizes on the estimation of the similarity metric. Each clause contains 1 slot. These datasets were re-generated for each run in our experiments so that each set would contain different clauses and we could test the generalizability of our similarity metric to unseen data. The corpus used for *supervised learning* is derived from 500 restaurant recommendations (from www.list.co.uk) containing the following slots: venue name, food type, area, and price range. See [30] for details on the corpus and annotations.

B. Results of Unsupervised Clause Clustering

In terms of unsupervised learning, we have applied the spectral clustering technique described in Section III-A. It relies on a set of numerical distances between clauses provided by four task-independent metrics: Meteor Score (MS), Word Recognition Accuracy (WRA), Semafor Semantic Score (SSS), and Semantic Textual Similarity (STS). These metrics were used because they provide affinities of lexical and semantic information. The motivation for using multiple metrics instead of a single metric is due to the lack of a task-independent metric providing meaningful distance scores between clauses. We compared the clustering accuracy (also referred to as 'purity') of lexical information (MS+WRA), semantic information (SSS+STS), and lexical plus semantic information (MS+WRA+SSS+STS), see Table I. Purity is computed as

$$Purity(C,S) = \frac{1}{N}\sum_{k} max_j |c_k \cap s_j|, \tag{7}$$

where $C = \{c_k\}$ is the set of clusters, $S = \{s_j\}$ is the set of slots, and N is the number of clauses. While bad clusterings have purity values close to 0, good clusterings have purity values close to 1. It can be observed that lexical information is better in the restaurant domain and semantic information is better in the movies domain. In contrast to other previous work limited to only semantic information [31], our results suggest that the combination of lexical and semantic information achieves the best results across datasets.

C. Results of Semi-Supervised Clause Clustering

In contrast to the previous Sub-section that relies on the similarity metric described by Equation 1, semi-supervised clustering relies on the induced similarity metric described by Equation 6. From Table II, we can note that a small amount of labelled data (e.g. 10 clauses) is sufficient to outperform purely unsupervised learning. However, it can also be noted that growing the dataset does not necessarily improve purity. This is presumably due to the fact that our Bayes nets overfit the given labelled data. Nonetheless, our results report that semi-supervised clustering is at least as good as unsupervised clustering for varying amounts of labelled data.

Metric	**Restaurants**	**Movies**
Random Selection	0.356	0.368
Lexical Information (MT+WRA)	0.470	0.404
Semantic Information (SSS+STS)	0.390	0.560
Lexical+Semantic Information	0.436	0.528

TABLE I. PURITY OF UNSUPERVISED CLAUSE CLUSTERING (AVERAGES OVER 10 RUNS) COMPARING LEXICAL AND SEMANTIC INFORMATION, SHOWING THAT ITS COMBINATION FINDS BETTER CLUSTERS THAN INDIVIDUAL METRICS IN ISOLATION ACROSS DATASETS.

Training Clauses	**Restaurants**	**Movies**
10	0.563	0.682
20	0.458	0.624
30	0.524	0.556
40	0.518	0.682
50	0.536	0.662

TABLE II. PURITY OF SEMI-SUPERVISED CLAUSE CLUSTERING (AVERAGES OVER 10 RUNS), SHOWING IMPROVED RESULTS OVER USING PURELY UNSUPERVISED LEARNING (SEE TABLE I).

D. Results of Supervised Semantic Slot Detection

In terms of supervised learning, we trained the Bayes net described in Section III-C from restaurant recommendations (from www.list.co.uk) including known slots based on 2900 training instances. This dataset was derived from labelled data of known slots (venue name, food type, area, price range). An evaluation on held-out data from known slots on a 10-fold cross validation reported a classification accuracy of 94.9%. To test the Bayes net's accuracy on unseen data, we tested it on the movies and restaurants data used for clustering. This attained the classification accuracies shown in Figure 4, where accuracy is computed as

$$Accuracy = \frac{TruePositives + TrueNegatives}{Positives + Negatives}. \tag{8}$$

While $Pr(label = yes|\mathbf{e}) \geq 0.5$ seems to be a reasonable threshold for detecting slots, some adaptation can be attempted (e.g. 0.2 in restaurants and 0.8 in movies) from the given initial data. This adaptation process is left as future work.

V. CONCLUSION AND FUTURE WORK

Recent work has trained semantic slot labellers with large amounts of labelled data [32] or no labelled data at all [31]. In this paper, we have addressed the problem of training a semantic slot labeller from minimally labelled data. Our learning algorithm uses semi-supervised clustering to identify sentences with similar semantics based on a learned Bayesian similarity metric from a small dataset taking lexical and semantic features into account. We have applied spectral clustering due to its robustness to variant cluster shapes. We have also applied a Bayes net to distinguish phrases that represented semantic slots from those that do not. Our experimental results

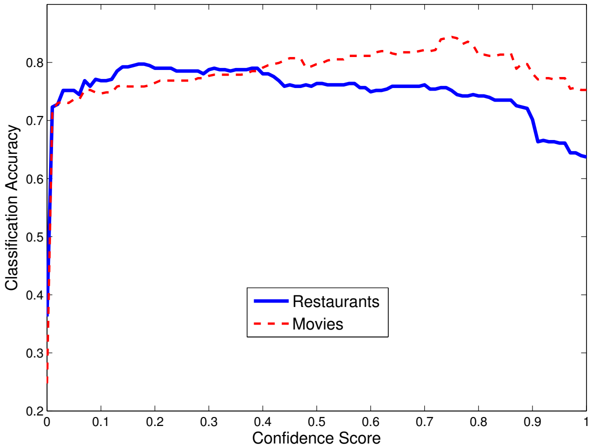


Fig. 4. Classification accuracies per confidence score (thresholds for accepting/rejecting hypotheses) in Bayesian semantic slot detection, where slots in the movies domain are easier to detect than in the restaurant domain.

in the domains of restaurants and movies report that: (i) semi-supervised clustering (based on a learned similarity metric) outperforms unsupervised learning; (ii) the previous result can be observed based on very few training data instances and that increasing the size of the training data does not lead to better performance; and (iii) lexical and semantic information contributes differently in different domains so that clustering based on both types of information offers the best generalisation.

In future work, we aim to (1) evaluate the proposed method with clauses including multiple slots; (2) evaluate the proposed method with larger sets of sentences and more scalable clustering methods [33]; (3) compare the proposed clause similarity metric with other metrics to assess generalisation in growing domains and across domains; (4) compare multiple unsupervised clustering methods and supervised classifiers using the method described above; an (5) perform an extrinsic evaluation with an end-to-end spoken dialogue system [34].

ACKNOWLEDGMENT

This research was funded by the European Commission FP7 programme FP7/2011-14 under grant agreement no. 287615 (PARLANCE).

REFERENCES

[1] S. Branavan, H. Chen, L. Zettlemoyer, and R. Barzilay, “Reinforcement Learning for Mapping Instructions to Actions,” in *ACL*, 2009.

[2] A. Vogel and D. Jurafsky, “Learning to Follow Navigation Directions,” in *ACL*, 2012.

[3] B. Snyder and R. Barzilay, “Database-Text Alignment via Structured Multilabel Classication,” in *IJCAI*, 2007.

[4] R. Barzilay and L. Lee, “Bootstrapping Lexical Choice via Multiple-Sequence Alignment,” in *EMNLP*, 2002.

[5] G. Angeli, P. Liang, and D. Klein, “A Simple Domain-Independent Probabilistic Approach to Generation,” in *EMNLP*, 2010.

[6] I. Konstas and M. Lapata, “Unsupervised Concept-to-Text Generation with Hypergraphs,” in *NAACL*, 2012.

[7] H. Cuayáhuitl, N. Dethlefs, H. Hastie, and X. Liu, “Training a statistical surface realiser from automatic slot labelling,” in *SLT*, 2014.

[8] W. Lu, H. T. Ng, W. S. Lee, and L. Zettlemoyer, “A Generative Model for Parsing Natural Language to Meaning Representations,” in *EMNLP*, 2008.

[9] P. Liang, M. Jordan, and D. Klein, “Learning Semantic Correspondences with Less Supervision,” in *ACL*, 2009.

[10] F. Mairesse, M. Gasic, F. Jurcícek, S. Keizer, B. Thomson, K. Yu, and S. Young, “Phrase-based statistical language generation using graphical models and active learning,” in *ACL*, 2010.

[11] M. MacMahon, B. Stankiewicz, and B. Kuipers, “Walk the Talk: Connecting Language Knowledge, and Action in Route Instructions,” in *AAAI*, 2006.

[12] T. Kollar, S. Tellex, D. Roy, and N. Roy, “Toward Understanding in Natural Language Directions,” in *HRI*, 2010.

[13] D. Chen and R. Mooney, “Learning to Interpret Natural Language Navigation Instructions from Observations,” in *AAAI*, 2011.

[14] P. Liang, *Semi-Supervised Learning for Natural Language*. Master’s thesis, Massachusetts Institute of Technology, 2005.

[15] S. Basu, A. Banerjee, and R. J. Mooney, “Active semi-supervision for pairwise constrained clustering,” in *SIAM*, April 2004.

[16] A. S. Gaard, *Semi-Supervised Learning and Domain Adaptation in Natural Language Processing*, 2nd ed. Morgan & Claypool, 2013.

[17] H. Daumé-III, “Frustratingly Easy Domain Adaptation,” in *Proceedings of the Association for Computational Linguistics (ACL)*, 2007.

[18] H. Daumé-III and D. Marcu, “Domain adaptation for statistical classifiers,” *Journal of Artificial Intelligence Research*, vol. 26, 2006.

[19] J. de Jesús Rubio, “Evolving intelligent algorithms for the modelling of brain and eye signals,” *Appl. Soft Comput.*, vol. 14, pp. 259–268, 2014.

[20] V. Punyakanok and D. Roth, “The use of classifiers in sequential inference,” in *NIPS*. MIT Press, 2001.

[21] U. von Luxburg, “A tutorial on spectral clustering,” *Statistics and Computing*, vol. 17, no. 4, 2007.

[22] E. Bair, “Semi-supervised clustering methods,” *CoRR*, vol. abs/1307.0252, 2013.

[23] M. Denkowski and A. Lavie, “Meteor universal: Language specific translation evaluation for any target language,” in *EACL Workshop on Statistical Machine Translation*, 2014.

[24] D. Das, N. Schneider, D. Chen, and N. A. Smith, “Probabilistic frame-semantic parsing,” in *HLT*, 2010.

[25] L. Han, A. Kashyap, T. Finin, J. Mayfield, and JonathanWeese, “UMBC EBIQUITY-CORE: Semantic textual similarity systems,” in **SEM*, 2013.

[26] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, “The stanford corenlp natural language processing toolkit,” in *ACL (System Demonstrations)*, 2014, pp. 55–60.

[27] G. Cooper and E. Herskovits, “A bayesian method for the induction of probabilistic networks from data,” *Machine Learning*, vol. 9, no. 4, 1992.

[28] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kaufmann, 2005.

[29] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, “Feature-rich part-of-speech tagging with a cyclic dependency network,” in *NAACL*, 2003.

[30] N. Dethlefs, H. W. Hastie, H. Cuayáhuitl, and O. Lemon, “Conditional random fields for responsive surface realisation using global features,” in *ACL*, 2013.

[31] Y.-N. Chen, W. Y. Wang, and A. I. Rudnicky, “Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing,” in *ASRU*, 2013.

[32] G. Tür, A. Çelikyilmaz, and D. Hakkani-Tür, “Latent semantic modeling for slot filling in conversational understanding,” in *ICASSP*, 2013.

[33] J. Liu, C. Wang, M. Danilevsky, and J. Han, “Large-scale spectral clustering on graphs,” in *IJCAI*, 2013.

[34] H. Hastie, M. Aufaure, P. Alexopoulos, H. Cuayáhuitl, N. Dethlefs, M. Gasic, J. Henderson, O. Lemon, X. Liu, P. Mika, N. Ben Mustapha, V. Rieser, B. Thomson, P. Tsiakoulis, Y. Vanrompay, B. Villazon-Terrazas, and S. Young, “Demonstration of the PARLANCE system: a data-driven incremental, spoken dialogue system for interactive search,” in *SIGDIAL*, Metz, France, August 2013.