Markov Chains for Robust Graph-based Commonsense Information Extraction

Niket Tandon\textsuperscript{1,4} Dheeraj Rajagopal\textsuperscript{2,4} Gerard de Melo\textsuperscript{3}

(1) Max Planck Institute for Informatics, Germany  
(2) NUS, Singapore  
(3) ICSI, Berkeley  
(4) PQRS Research, pqrs-research.org  
ntandon@mpi-inf.mpg.de, tsldr@nus.edu.sg, demelo@icsi.berkeley.edu

Abstract

Commonsense knowledge is useful for making Web search, local search, and mobile assistance behave in a way that the user perceives as “smart”. Most machine-readable knowledge bases, however, lack basic commonsense facts about the world, e.g. the property of ice cream being cold. This paper proposes a graph-based Markov chain approach to extract commonsense knowledge from Web-scale language models or other sources. Unlike previous work on information extraction where the graph representation of factual knowledge is rather sparse, our Markov chain approach is geared towards the challenging nature of commonsense knowledge when determining the accuracy of candidate facts. The experiments show that our method results in more accurate and robust extractions. Based on our method, we develop an online system that provides commonsense property lookup for an object in real time.

KEYWORDS: commonsense knowledge, knowledge-base construction.
1 Introduction

For a search query like “what is both edible and poisonous?”, most search engines retrieve pages with the keywords edible and poisonous, but users increasingly expect direct answers like pufferfish. If a user wants hot drinks, a mobile assistant like Siri should search for cafés, not sleazy bars. Commonsense knowledge (CSK) has a wide range of applications, from high-level applications like mobile assistants and commonsense-aware search engines (Hsu et al., 2006) to NLP tasks like textual entailment and word sense disambiguation (Chen and Liu, 2011).

State-of-the-art sources like Cyc and ConceptNet have limited coverage, while automated information extraction (IE) typically suffers from low accuracy (Tandon et al., 2011), as IE patterns can be noisy and ambiguous. Large-scale high precision IE remains very challenging, and facts extracted by existing systems thus have rarely been put to practical use. Previously, Li et al. (Li et al., 2011) filtered facts extracted from a large corpus by propagating scores from human seed facts to related facts and contexts. However, their method does not handle the very ambiguous patterns typical of CSK. FactRank (Jain and Pantel, 2010) uses a simple graph of facts to find mistyped or out-of-domain arguments. However, they do not exploit the large number of seeds provided by databases like ConceptNet for robust pattern filtering.

In contrast, our work proposes a joint model of candidate facts, seed facts, patterns and relations geared towards Web-scale CSK extractions. Standard IE deals with fact patterns like \(<X> is\ married\ to\ <Y>\) that are fairly reliable, so the same tuple is rarely encountered for multiple relations simultaneously and the resulting graphs are sparse. Our approach instead deals with CSK IE. Owing to the much more generic nature of patterns, e.g. \(<X> is/are/can\ be\ <Y>\), an extracted tuple frequently has more than one candidate relation, leading to much more challenging graphs. This paper addresses these challenges with a graph-based Markov chain model that leverages rich Web-scale statistics from one or more large-scale extraction sources in order to achieve high accuracy. We develop an online system that can lookup commonsense property knowledge in real time. We make use of anchored patterns for fast lookup. The system provides a ranked property tag cloud. These scores are obtained using our graph-based Markov chain model.

2 Approach

Tuple Graph Representation. We follow the standard bootstrapped IE methodology, where seed facts lead to patterns, which in turn induce newly discovered candidate facts (Pantel and Pennacchiotti, 2006). We apply it, however, to Web-scale N-Grams data. For a given candidate fact \(T\), a directed tuple graph \(G_T = (V, E)\) is created. The node set includes nodes for the candidate tuple \(T\), for the pattern set \(P\) that extracted \(T\), for the seed set \(S\) that induced \(P\), as well as all for the relations \(R\) that \(s \in S\) belong to, plus an additional artificial relation node \(\text{NO\_RELATION}\) to account for noise. A weighted edge from the tuple node \(v_t\) to one or more patterns \(v_p\) corresponds (after normalization) to a tuple probability \(Pr(p|t)\), which is estimated using the pattern frequency in an extraction source like the Google N-grams dataset. The outgoing edge weights of a node are normalized to sum to 1 in order to give us such probabilities. A weighted edge from a pattern node \(v_p\) to one or more seed nodes \(v_s\) corresponds to \(Pr(s|p)\) probabilities, which are estimated as the seed confidence scores delivered by the source of the seeds (ConceptNet in our case). These, too, are normalized as above. A weighted edge from a seed node \(v_s\) to a relation node \(v_r\) corresponds to the conditional relation probability \(Pr(r|s)\), and is estimated as 1 over the number of relations a seed belongs to. A weighted edge from every node to \(\text{NO\_RELATION}\) with the edge weight proportional to the average
of outgoing edges of the node and inversely proportional to the number of outgoing edges as \( \sum_{i} \frac{w_{out,i}}{\text{num}_{\text{out}}} = k_{1}\text{num}_{\text{out}}^{k_{2}} \), where \( k_{1} \) and \( k_{2} \) are parameters. The first component provides scaling and the second component depicts lower chances of noise when the tuple matches several patterns or when a pattern is generated from several seeds. All other node pairs remain unconnected in the adjacency matrix. Figure 1 shows an example of a tuple graph. One of the desirable properties of this model is that it is localized, allowing us to consider the local graph \( G_{T} \) for each tuple instead of a single graph with potentially millions of seeds, patterns and tuples. Our method can thus be parallelized very easily without additional communication overhead.

**Fact scoring and classification.** The Markov chain consists of states for every node and a transition matrix \( Q \). The state transition probabilities in \( Q \) are fixed by taking the edge weights and 1) incorporating random transitions to the \( \text{NO RELATION} \) state in order to account for uncertainty and noise, and 2) additionally incorporating a random restart jump from any \( v \in V \) to \( v_{i} \) with probability \( \alpha \) (instead of the unmodified transitions with probability \( 1 - \alpha \)), in order to satisfy ergodicity properties. One can then prove that a random walk using \( Q \) has a well-defined and unique stationary distribution over the nodes. The stationary probability of being at a certain relation node can be leveraged to classify the relation that the tuple belongs to, along with the confidence of classification. When the tuple matches few patterns it is likely to be noisy. For such tuple graphs, \( \text{NO RELATION} \) has a higher stationary probability because the other edges carry low weights. We use the standard power iteration method to compute the stationary distribution using \( Q \). Upon convergence, we determine the relation node \( v_{i} \) whose stationary distribution is the highest. If this is the \( \text{NO RELATION} \) node, the fact is treated as noise and rejected.
3 System

The system takes as input a common noun like car, flower and provides a scored list of commonsense properties with visualization. The first step involves constructing semi-instatiated patterns (SIP) from the input, i.e. anchoring the input with patterns. For example, car/NN is very */JJ, car/NN seems really */JJ are some SIPs in the current example. These SIPs are looked up in three resources: Google N-grams corpus, Wikipedia full text and Microsoft N-grams.

For fast lookup, we construct a SIP online lookup system:

- An index lookup over Wikipedia: Wikipedia XML dump was converted to text format and indexed in Lucene with stop-words included. We developed a lookup service takes a SIP as input and fetches relevant text by looking up the index for fast retrieval required for our online system.

- An index lookup over Google N-grams corpus: Google has published a dataset of raw frequencies for n-grams ($n=1,\ldots,5$) computed from over 1,024G word tokens of English text, taken from Google’s web page search index. In compressed form, the distributed data amounts to 24GB. We developed an index lookup service over Google n-grams such that given a SIP, all relevant Google 5-grams are fetched. 5-grams provide the largest context and are therefore preferred over 4-grams.

- Bing N-grams Web service caller: Microsoft’s Web N-gram Corpus is based on the complete collection of documents indexed for the English US version of the Bing search engine. The dataset is not distributed as such but made accessible by means of a Web service described using the WSDL standard. The service provides smoothed n-gram language model based probability scores rather than raw frequencies (Wang et al., 2010). We enable SIP lookup over this service.

The results from these resources are then merged to generate tuple statistics of the form: $x,y,[pattern:score]$. These statistics form the input to our Markov chain method which provides a scored list of facts which are then displayed. Due to the locality of our approach, the system operates online. Figure 2 shows the flow of the system.

Figure 3,4 provides screenshot for the output of property lookup for flower and fish at pattern support 2.

4 Experiments

4.1 Experimental Setup

Using ConceptNet, we obtain patterns of the form $X_{NN}$ is very $Y_{JJ}$, where the subscripts are part-of-speech tags, $X$ is the subject, $Y$ is the object and is very is the pattern predicate. We retrieve the top patterns for a relation extracted from ConceptNet, sorted by frequency (i.e., how often it is present in ConceptNet’s OMCS sentences). Such patterns may sometimes be too generic (e.g. $<X>$ are $<Y>$), and lead to noise during fact extraction, but the model accounts for this. For tuple extraction, our primary source of extraction (RW1) are the Google 5-grams data.
4.2 Experimental results

The gold set consists of 200 manually classified facts, 100 negative and 100 positive. The parameter selection for NO_RELATION edge weight is performed over F1 score. The best parameters are obtained at small k1 and large k2 values, see Figure 5. The minimum pattern support is 2, i.e. candidate assertions with less than 2 patterns matched are dropped because they have insufficient evidence.

As baseline, we consider the reliability of a tuple \((r_i)\) using the state-of-the-art modified Pointwise Mutual Information (PMI) by (Pantel and Pennacchiotti, 2006). Table 1 reports the evaluation results. The markov chain approaches significantly gain accuracy over the baseline.
### Conclusion

We have presented a novel approach for joint commonsense information extraction. Our method shows clear improvements over several commonly used baselines and can easily be integrated into existing information extraction systems. We have applied our algorithm within a larger setup that also incorporates Web-scale language models. Together, this framework allows us to extract large yet clean amounts of commonsense knowledge from the Web.
References


