Combining Neural Networks and Pattern Matching for Ontology Mining - a Meta Learning Inspired Approach

Dmitri Roussinov
Department of Computer and Information Sciences
University of Strathclyde
Glasgow, United Kingdon, G11XQ
Email: dmitri.roussinov@strath.ac.uk

Nadezhda Puchnina
University of Tallinn
Tallinn, Estonia, 10120
Email: nadinpuchnina@gmail.com

Abstract*—Several applications dealing with natural language text involve automated validation of the membership in a given category (e.g. *France* is a *country*, *Gladiator* is a *movie*, but not a *country*). *Meta-learning* is a recent and powerful machine learning approach, which goal is to train a model (or a family of models) on a variety of learning tasks, such that it can solve new learning tasks in a more efficient way, e.g. using smaller number of training samples or in less time. We present an original approach inspired by meta-learning and consisting of two tiers of models: for any arbitrary category, our *general model* supplies high confidence training instances (seeds) for our *category-specific* models. Our general model is based on pattern matching and optimized for the precision at top N, while its recall is not important. Our category-specific models are based on recurrent neural networks (RNN-s), which recently showed themselves extremely effective in several natural language applications, such as machine translation, sentiment analysis, parsing, and chatbots. By following the meta-learning principles, we are training our highest level (general) model in such a way that our second-tier category-specific models (which are dependent on it) are optimized for the best possible performance in a specific application. This work is important because our approach is capable of verifying membership in an arbitrary category defined by a sequence of words including longer and more complex categories such as *Ridley Scott movie* or *City in southern Germany* that are currently not supported by existing manually created ontologies (such as Freebase, Wordnet or Wikidata). Also, our approach uses only raw text, and thus can be useful when there are no such ontologies available, which is a common situation with languages other than English. Even the largest English ontologies are known to have low coverage, insufficient for many practical applications such as automated question answering, which we use here to illustrate the advantages of our approach. We rigorously test it on a number of questions larger than the previous studies and demonstrate that when coupled with a simple answer-scoring mechanism, our meta-learning-inspired approach 1) provides up to 50% improvement over prior approaches that do not use any manually curated knowledge bases and 2) achieves the state of-the-art performance among all the current approaches including those taking advantage of such knowledge bases.

I. Introduction

While recent advances with distributed representations allowing efficient back-propagation have been behind many breakthroughs in natural language applications, computer vision and AI in general [1], it has been also noted that the tasks of capturing common-sense semantic relationships remain largely unsolved, among them the task of verifying that a given pair of concepts represents a valid category-instance relation: *(country, France), (movie, Gladiator), (Las Vegas hotel, MGM Grand)*, but not *(country, Gladiator)*. This task is often needed in **question answering**, when the correct answer to the question (e.g., *What Las Vegas hotel was made famous by the Rat Pack?*) is expected to belong to a certain category (e.g., *Las Vegas hotel*). **Information retrieval** often benefits from expanding original user query with hyponyms (words with more specific meanings), e.g. *transportation disasters* $\rightarrow$ *railroad disasters*. Natural language processing tasks such as **parsing, relation extraction, anaphora and co-reference resolution** also benefit from knowing the semantic properties of the nouns, e.g., if they are animate (can refer to living beings) or not. For the task of **database federation**, an attribute in one database (e.g., with values *France, Germany,* and *UK*) often needs to be automatically matched with an attribute called *country* in another database. The task of information extraction is often framed as identifying instances of a specific subset of categories of interest, such as *person, organization, location,* etc.

There are several other closely related tasks to automated semantic category verification, often sharing common approaches and datasets, including *ontology building, automated ontology population, taxonomy mining, "is a" or hyponym/hypernym relation verification* or *information extraction*. The task is also believed to be fundamental to human cognition [2].

While many successful applications use manually curated or crowdsourced resources such as Freebase or WordNet, here we focus on automated validation that can make use of only raw (unstructured) text. This is for several reasons: 1) We believe those two types of approaches are complementary to each other and both deserve researchers' attention. 2) Manually created ontologies are often contradictory or incomplete [3][4]. For example, the creators of Watson, an IBM computer winning several trivia competitions, noted that Freebase covered only 20% of entities mentioned in Jeopardy questions [5]. A brief look at categories occurring in popular test questions (TREC [6] or WebQuestions [7]) also reveals that most of them are not present in even the largest manual ontologies, especially "composite" categories such as *Ridley Scott movie* or *city in southern Germany*. 3) Most languages in the world

simply do not have such manual resources, while still having plenty of raw text, including that on the Web and in social media.

Our approach allows to validate any arbitrary pair "on demand" rather than trying to gradually assemble a large ontology in approaches such as in [8][9]. While prior research has extensively looked at the accuracies of various ontology mining approaches by comparing against manually created ones held as gold standard, here we chose a different, often overlooked by the prior works path: we look at the verification performance in a specific "downstream" task, which is in our case open domain factoid question answering [6], e.g. *What Ridley Scott movie is set in 180 a. d.?* Answer: *Gladiator*. We have specifically looked at the questions that expect the answer to belong to a certain category, e.g. *movie* or *Las Vegas hotel*.

Our approach has been inspired by recent advances in *meta-learning* [10]. We approach the problem by considering *two-tier* models: 1) A *general model* that has been trained on a sample of categories and optimized for the precision at top N. Our general model works in a fully automated way and provides high-confidence training examples ("seeds") for any category defined by an arbitrary sequence of up to 4 words. 2) Those seeds ("low hanging fruits") are used as training examples for *category-specific* models. The entire model is trained as *learning to learn* the category-specific verification so the overall answer accuracy is maximized. We use a pattern-matching approach [11] for the general model as it is known to provide high precision, sometimes at the expense of recall. We use a *recurrent neural network* (RNN) for our category-specific models. RNN-s showed themselves very effective in most natural language applications, such as machine translation, sentiment analysis, parsing, and chatbots [1] and are behind most state-of-art approaches to those tasks. Our 2-tier combination is effective since it allows to feed on much larger data, compared to a more traditional "single-tier model", due to automatically identifying training seed instances, and considering the text segments in which they occur.

Our contributions are the following: 1) We show that when coupled with a simple answer scoring mechanism, our metalearning-inspired semantic verification delivers up to 50% improvement relatively to the other known approaches capable of validating membership for an arbitrary (not necessary existing in a knowledge-base) category-instance pair. 2) Our performance is comparable with the best systems including those that do take advantage of manually developed knowledge-bases, thus suggesting that raw text mining can be used as a replacement or a complement for them. 3) We have resolved several technical issues in applying a recurrent neural network (RNN) to validate membership in a category, where the candidates are represented by a sample of text segments in which they occur. 4) We have also empirically confirmed prior observations that even the largest crowdsourced or mined ontologies are grossly incomplete for the question answering task. When used directly, without applying additional approximate matching or reasoning algorithms, they fall short of providing the state-of-the-art performance.

The next section overviews the prior related work. It is followed by the description of our framework suggested here, followed, in turn, by our empirical results. The "Conclusions, Limitations and Future Work" section summarizes our findings.

II. Related Work And Baselines

A. Semantic Category Verification

Semantic verification and closely related tasks have a long history. First matching patterns in a verification corpus is typically attributed to Hearst [11]. Work by Brin [12] presented a pattern-based bootstrapping approach. Variations of the approach have been suggested by other researchers: for anaphora resolution [13], adapted in specific domains [14], and checking causation [15]. In addition to smaller local corpora, the patterns were also matched on the Web [16][17]. Yates and Etzioni [18] developed a probabilistic model by building on the classic balls-and-urns problem from combinatorics and created KnowItAll text mining tool to automatically collect relationships, including the hyponymic from the Web. It was evaluated by recall and precision on four categories only: *Corporations, Countries, CEO of a company*, and *Capital of a Country*. They used mutual information between the category and the instance as the primary validation metric, which we also report here as one of the baselines. Schlobach et al. [19] studied larger number of categories, but limited to geography domain. Igo & Riloff [20] used a bootstrapping algorithm and co-occurrence statistics between each lexicon entry and semantically related terms. Their approach was evaluated on 7 semantic categories representing two domains, and required seed words for each semantic category. Fleischmann and Hovy [21] used 8 classes (*athlete, politician/government, clergy, businessperson, entertainer/artist, lawyer, doctor/scientist, police*) and after examining several machine learning algorithms reported the best accuracy of 70.4%. Collobert and Weston [22] first used a deep neural network, combined with supervised multitask learning to verify various types of semantic relations, not specifically targeting category verification, and achieving 70-80% accuracy when the labeled data is available for the relation.

The above mentioned approaches have been extensively benchmarked on how they resemble manual (crowdsourced) ontologies. However, the latter are known to be grossly incomplete for many real-life applications [3][4][5]. Thus, we argue that it is worth evaluating both manual and mined ontologies in the context of specific downstream applications, which still remains however limited. [23] explored how an arbitrary (not pre-anticipated) category can be validated in order to help factoid question answering, but their reported performance was below the approaches replying on manually provided training data. [23] introduced a model based on pointwise mutual information which considered absences of occurrences of certain patterns in a validation corpus as well. The approach was evaluated on a set of 109 test questions with 51 unique categories.

Among the noticeable automated ontology building projects are NELL [8] by CMU and MCG [9] by Microsoft Research. However, as we illustrate below in our empirical section, their

category coverage remains too low for the task considered here. We are also not aware of any experiments testing those systems for any practical down-stream task.

While a number of recent works based on deep learning targeted discovering semantic relations from word embeddings obtained by neural language models (e.g. [24]) and are capable of supporting "semantic arithmetic" like *king - man = queen - woman*, they are limited to single-word concepts. As a result, many named entities remain "out of vocabulary" when needed by a particular downstream task. Levy et al. [25] demonstrated that purely distributional (word embeddings-based) approaches tend to learn how likely individual words are to be possible category names or instances of some category, rather than relating them as a correct category-instance pair. Shwartz et al. [26] successfully combined both distributional and path-based approaches into one framework that uses a Long-Short Term Memory Network and parsed Wikipedia as a validation corpus. Somewhat similar to our verification approach was explored in [27] but focused on automatically evaluating negative examples for boot-strapping and tested within a narrow domain. Still, *the task of totally automated semantic verification in an arbitrary category, along with a more general task of capturing common sense knowledge, remain largely unsolved*.

B. Factoid Question Answering

Since our goal here is not to improve question answering (QA) algorithms per se, but rather use it as a possible down-stream application which dictates the distributions of categories and candidates to check, here we only mention works that shed light on what is the state-of-the art performance in the settings similar to ours. We approach open domain factoid QA [6] task, finding the correct answer to *What business was the source of John D. Rockefeller's fortune?*, which is *oil*. The answer is a simple fact, so typically no longer than 4 words, and is expected to be found in a fixed corpus (Wikipedia in our settings, while *Aquaint* originally in [6]). Various other settings exist in research, e.g. when the answer is guaranteed to come from a given short text segment [28], which is also referred as "comprehension" test.

The performance of each approach greatly depends on 1) the sample of test questions, 2) the information sources, and 3) the amount of knowledge-engineering involved. There are no standard universal benchmarks for all types of questions. Here, we only look at the questions that explicitly state the semantic category of the expected answer, e.g. *business* in the question above.

The settings similar to ours were first used by McNamee et al. [29] who applied a dependency parser and manual labeling to test the effect of semantic verification on a subset of 242 TREC questions, using Wikipedia, Aquaint and TREC 4-5 corpora as the answer sources. They reported the accuracy (proportion of correctly answered questions) of 30%. Since they did not list the questions used, we cannot use their result here for direct comparison.

A more recent work by Bordes et al. [30] involved several large knowledge bases including Freebase and reported the accuracy of 60-70% on similar types of questions and Wikipedia as the source of answers. The work by Chen et al. [31] used deep neural architecture, Wikipedia as the source, and a larger set of TREC factoid questions reporting around 40% accuracy on all types. Their work also referred to several other systems performing in the similar range when tested in the similar settings. Similar questions were also used to evaluate pattern-based approaches such as those based on AskMSR+ and their predecessors, reporting the accuracy around 60% [32][33].

[7] reported the accuracy of 53% on a similar WebQuestions dataset. They suggested a way to answer questions based on an existing Knowledge Base (KB) via approximate reasoning (probabilistic predicate inference). The authors do not provide the details how their KB has been built. However, there are related works by the same authors (e.g. [34]) that look at automated ways to build a KB, thus their KB in [7] may be at least somewhat automatically built, so their result can be also indicative of possible state-of-the-art performance.

While IBM's Watson [5] won several trivia competitions and seemed to demonstrate the answer accuracy above 90%, the details on its components and performance remain a trade secret. Besides, it relied on specialized hard-ware and was optimized for trivia questions. Thus, the *accuracies reported by the works mentioned above suggest that the state-of-the-art performance on the TREC (and similar) factoid questions, with an explicitly specified category of the answer, and using Wikipedia as the source, varies between 40% and 70%.*

C. Meta-learning

For a more comprehensive review of meta-learning, we are referring the reader to [10]. In brief, the goal of meta-learning is to train a model (or a family of models) on a variety of learning tasks, such that it can solve new learning tasks in a more efficient way, e.g. using smaller number of training samples or in less time. For example, [35] proposed a model-agnostic meta-learning approach to train the parameters of a deep neural model explicitly such that a smaller number of gradient steps is needed. They demonstrated the value of their approach on image classification and imitation-based robot training using a single demonstration. In our work, we are training our highest level (general) model in such a way that our second-tier category-specific models (which are dependent on it) are optimized for the best possible performance. The next section provides more details.

III. Two-Tier Neural Semantic Validation

A. Overall Architecture

We follow a two-tier approach: 1) Our *general model* is expected to provide a certain number (10 in this study) of reliable training examples for any arbitrary category. The recall of our general model is not important. For example, our general model does not need to accurately validate every possible candidate (word sequence) as being a *color* or not, but is instead expected to identify 10 reliable examples of colors. 2) *Category-specific models.* A separate model for any category C of interest (e.g. *color*) is trained fully automatically, on demand, using the training examples ("seeds") provided by the general model (e.g. *red, green, orange*). Randomly sampled seeds from other categories act as negative examples. As a result, a category-specific model is capable of validating

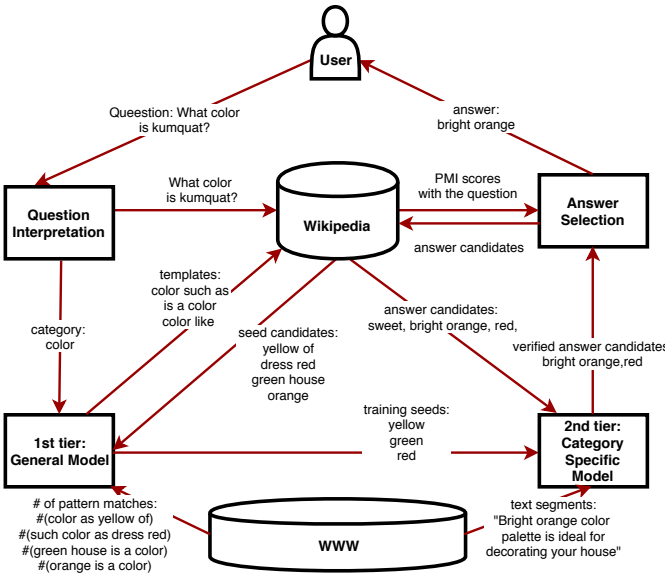

Fig. 1. A high-level overview of the data flow while answering a question.

other possible instances of C (e.g. *ash-gray*), even when the general model can not make a confident decision on them. Our category-specific models are based on Recurrent Neural Networks.

Figure 1 presents a high-level overview of the data flow while performing semantic verification to answer the question *What color is kumquat?* along with the examples of the training seeds and the answer candidates involved. The following subsections provide more details.

B. General Model

1) Sources of Instances: The inputs to our general model (called *seed candidates*) are obtained by searching a large corpus (Wikipedia here) for occurrences of the words representing the category followed by certain linguistic markers (we used "such as", "like", and "called", e.g. "military rank such as" and considered all sequences up to 4 words from the same sentence), until the desired number of candidates is found (1000 in our experiments). Although the specific markers to use may depend on a particular language used (English, French, Russian, etc.), all known modern languages have them. This approach is similar to classical Hearst-patterns [11], and provides precision ranging from 1% to 30% depending on the category. Obviously, such accuracy is often not sufficient to use the candidates as training seeds, thus, they need to be filtered further. This is exactly what our general model accomplishes.

2) Custom Training Dataset: We needed it since 1) we sought to avoid dependence on pre-existing ontological resources such as Freebase or WikiData, for the reasons presented in our introduction and 2) since none of them provides necessary recall, e.g. the lists of all *films* or *music bands* are typically only 50% complete. If using them, we would potentially face up to 50% mislabeling rates which would make it impossible to target 90%+ precision of training examples normally provided by humans. However, since we needed to check only top 10 candidates for each category, the time to create our dataset was relatively trivial (approximately 30 person-hours) compared with the total time invested in this project, which included designing and implementing the models, testing them and producing the manuscript for submission. Once our general model is trained, it is applied in a totally automated way to any arbitrary category.

3) Verification Model: We use the model from [36] because it was introduced for the similar purpose (to help question answering) and has demonstrated good generalization from the categories which it has not seen during training, as long as a large validation corpus is available such as a snapshot of World Wide Web or its n-grams statistics. The model is different from the preceding pattern-based models by 1) taking the absences of matches to validation patterns as well as their existence 2) accounting for those absences that are likely due to a limited size of the validation corpus.

Similar to the preceding works, the model uses features derived from pointwise mutual information (PMI), which captures non-randomness of the occurrence of a certain pattern. Specifically, for each validation pattern $p = a + b$, the model defines

$$I_{PMI}(a+b) = \pi(a+b) = \frac{\#(a+b)}{\#a \cdot \#b} \tag{1}$$

where a and b are the constituent parts of the pattern. E.g., to validate that *Microsoft* is a *company*, the pattern *company such as Microsoft* is segmented as *company such as* + *Microsoft*. $\#(p)$ is the number of matches to the pattern p in the corpus. As in [36], we use the simplest pattern language that does not involve any information on the part of speech, dependency, grammatical or semantic parsing, while the model is capable of including that as well. Thus, each pattern match is simply an exact string match. While this may not provide a high recall, especially for the pairs where the instance and the category occur several words apart, our general model is only expected to identify several reliable "seed" instances, thus is precision oriented. The high precision is achieved by combining the scores from several patterns, as explained in the following.

When matching in a limited size corpus, even such a large one as the entire indexed part of the World Wide Web, many patterns do not produce any matches. This results in some undefined PMI metrics. In order to deal with this type of undefined data, *the model operates with the estimated upper and lower bounds of PMI metrics rather than with the metrics themselves* as defined in the following. By approximating the distribution of $\#(p)$ by the Poisson distribution (commonly used for word counts in a corpus, e.g. [37]) , we estimate its standard deviation as its square root:

$$\sigma(\#p) = \sqrt{\#p} \tag{2}$$

Next, we model defines the *upper bound estimate* for $\#(p)$ as follows:

$$\overline{\#p} = \begin{cases} \#p + \sqrt{\#p} & \text{if } \#p > 0 \\ 1 & \text{if } \#p = 0 \end{cases} \tag{3}$$

The *lower bound estimate* for $\#(p)$ is defined similarly, while the correction is made in the opposite direction:

$$\underline{\#p} = \#p - \sqrt{\#p} \tag{4}$$

The corrections above are only noticeable for small $\#(p)$, e.g. for $\#(p)$=10000 the relative correction is only around 1%. Next, the model defines the upper and low bound estimates for the PMI metric as the following:

$$\overline{\pi(a+b)} = \frac{\overline{\#(a)}}{\underline{\#a} \cdot \underline{\#b}} \tag{5}$$

$$\underline{\pi(a+b)} = \frac{\underline{\#(a)}}{\overline{\#a} \cdot \overline{\#b}} \tag{6}$$

A low value for the estimate of the upper bound $\overline{\pi(a+b)}$ serves as a signal that a certain pattern likely occurs only due to a random chance and, thus, it serves as an indication that the category membership is unlikely. Conversely, a high value estimate of the lower bound $\underline{\pi(a+b)}$ signals that the non-randomness of occurrence is strong and the membership is very likely.

To improve generalization across different categories, the model converts the above estimates into boolean variables by comparing them against the upper and low bounds with certain thresholds:

$$\overline{b_{a+b}} = \overline{\pi(a+b)} > \overline{t(p)} \tag{7}$$

$$\underline{b_{a+b}} = \underline{\pi(a+b)} < \underline{t(p)} \tag{8}$$

And the thresholds are set to the upper and lower bounds of the average PMI scores computed for a random sample of candidates for the same category:

$$\overline{t(p)} = \mathrm{E}[\pi(p)] + \sigma(\pi(p)) \tag{9}$$

$$\underline{t(p)} = \mathrm{E}[\pi(p)] - \sigma(\pi(p)) \tag{10}$$

where $\mathrm{E}[\pi(p)]$ is the average and $\sigma(\pi(p))$ is the standard deviation of $\pi(p)$. Those boolean variables effectively act as neurons which can fire or not depending on the above PMI values. The adjustments above are crucial for the model to generalize to new unseen categories.

We use up to 52 patterns, representing various known linguistic markers (most taken from [36]), e.g. *movies such as X, such movies as X, movies like X, X is a move, X the movie, the movie X,* etc. The boolean features are combined by a logistic regression. While the choice of validation patterns is pre-set in our general model, in a future more general approach, they can be learned similarly to convolutional filters popular in deep learning networks. We used Microsoft Bing n-grams service [38] to obtain the numbers of pattern matches.

C. Category Specific Models

1) Instances to Validate: The set of candidate instances to which the category-specific models are applied is dictated by a particular downstream application, which in our case is open domain factoid question answering, as further explained below under "Question Answering Mechanism". The instances that need to be verified are possible answers being considered (called answer candidates). E.g. *What color is kumquat?* may result in the candidates *sweet, orange, and red*, out of which *sweet* does not validate to the expected category of *color* so is excluded.

2) Text Segments Representing the Instances: Each category-instance pair (e.g. *color-red*) is represented by a set of text segments, from a certain validation corpus, in which both of them occur together in the same sentence, e.g. "Red, the color of blood and fire, is associated with meanings of love." The segments are symmetrically truncated to match the maximum size of the recurrent neural network that receives them as inputs (10 words here) and to provide higher generalization. The duplicate segments are removed upon that.

To provide generalization, the words representing the category are replaced with a special marker (e.g. *<cat>*). During training, the words representing the positive and negative seeds are also replaced with another marker (e.g. *<seed>*). For example, the segment *such color as green* becomes *such <cat> as <seed>*. When applying the trained model, the words representing the candidate instances are replaced with the same marker (*<seed>*). For example, *such color as ash-grey* would also become *such <cat> as <seed>*, and thus scored by our RNN the same way. The crucial difference between using RNN and pattern matching, is that the latter can learn to provide similar scores to similar patterns, thus effectively acting as trainable by back-propagation "soft" (approximate) template matching mechanism.

Our preliminary results and those reported here suggest that using Wikipedia as a validation corpus is not sufficient, so we obtained the segments by running a query consisting of the category and the instance (e.g. *color red*) through Microsoft's Bing search engine, and extracting the segments from the first 400 results. The order of results is not important, and they are shuffled before used. The category-instance pairs with fewer than 10 segments found this way are classified as false since that normally suggests they are not related.

3) RNN GRU Binary Classifier: We use a recurrent neural network (RNN), specifically the one with gated recurrent units (GRU) [39], which is similar to LSTM (Long Short-Term Memory) network. Those two types of networks with similar properties have been behind many recent advances in machine translation, sentiment analysis, and dialogue modeling, and currently deliver state of the art performance in those tasks. They are capable of automatically learning the properties of individual words and patterns of their use, in order to perform binary classification of text segments (as here), or to map word sequences to other sequences (as in machine translation). Convolutional Neural Networks can be potentially used as well, but we left that for future research.

Our RNN classifier is trained to predict whether a given text segment is coming from a positive seed (provided by the general model) or from a negative seed (randomly sampled from positive seeds of other categories). Thus, we feed the segments coming from the positive seeds labeled as 1, and those coming from the negative seeds as 0. We set the learning rate of 0.03 for backpropagation, typical for RNN applications to text. We use the cross entropy as our loss function. A random sample of 1000 segments is removed from the training set and used as a validation (development) set: after 200 epochs, the best performing model, as measured on this validation set, is chosen as the final trained model.

The number of segments to train each category specific model ranged from 1,000-s to 10,000-s (depending on the number of segments found), and was taking 22 seconds in average to train.

Once trained, our RNN is used to score the segments of the candidates that we need to validate. Those scores are averaged for each candidate. Only the candidates with the average scores above certain threshold are classified as true (valid) instances. The threshold is automatically set for each category such as to reach the desired precision (90% in our experiments) as measured using the same training seeds.

During our preliminary experiments we settled on the following configuration for our GRU network: a single layer, with the context dimensionality of 30, word embedding dimensionality also of 30, and the maximum length of 10 words. While most NLP applications typically involve the embeddings dimensions of 50-1000, we found that larger than 30 dimensions did not provide additional benefits, possibly since the lexical and grammatical diversity in our segments is low compared with the applications where entire sentences are processed.

4) Question Answering Mechanism: Practical impact of semantic verification can only be claimed with respect to a certain application that uses it. This application determines the sample of category-instance pairs that need to be verified. We used open domain factoid question answering (QA) [6], where the answer is sought in a given corpus (Wikipedia here). Since our primary objective here was to provide a downstream application which determines the sample of category-instance pairs, but not to improve QA as such, we sought a simplest answer finding algorithm for our question types. Thus, we settled on an algorithm inspired by the prior research on redundancy-based question answering [32][33], which provides comparable with the state of the art accuracy. Simply speaking, for each question Q with the stated answer category C, our algorithm returns a valid instance of C that is most highly associated with the words in Q. This is accomplished by the following sequence of steps: 1) Our QA algorithm limits the search to top 1000 sentences scored by TF-IDF formula using the question as the query. 2) All the sequences of up to 4 words (4-grams) from those sentences and their immediate neighbors are considered as candidate answers. 3) the candidates are ranked by the strength of their association with the words in the question. The strength is measured as pointwise mutual information between the candidate (as a phrase) and the question (all words from the question combined by boolean AND, stopwords removed) computed on the entire answer corpus (Wikipedia). The highest scoring candidate that correctly validates to the desired category by the category-specific model is chosen as the final answer to the question. For example, for a question *What Ridley Scott movie is set in 180 a. d.?* out of top 1000 candidates only the following correctly validate to the expected category: *American Gangster, Blade Runner,* and *Gladiator*. But since "180" co-occurs only with *Gladiator*, the correct answer wins.

TABLE I
GENERAL MODEL. PRECISION AT 10 ON THE TRAINING AND TESTING SETS.

Configuration	Train	Test
Our Models:		
Entire Web statistics	89%	83%
Only Wikipedia statistics	72%	68%
Baselines from prior works:		
PMI association between the category and the instance from [18]	39%	39%
Co-occurrence patterns from [23]	59%	56%

IV. EMPIRICAL EVALUATION

A. Data Sets

We tested our approach on subsets of well known CuratedTREC [40] and WebQuestions [7] datasets. CuratedTREC includes most of the questions from TREC Question Answering competition over several years [6]. WebQuestions was created by crawling questions through the Google Suggest API, and then obtaining answers using Amazon Mechanical Turk. We only used the questions that explicitly state the semantic category of the answer. We decided not to use the implied categories such as "who" (person), "where" (location) or "when" (date).We also did not use the questions expecting a number. Thus, to keep our selection mechanism simple and reproducible, we preserved only the questions matching the regular expression $(What|Which)(.+)(do|does|did|is|was|are|were)$ and randomly sampled *100* questions for training and *1000* questions for testing. The training and testing sets do not share any common categories of the expected answer. Those numbers allowed us to keep the labeling (creating the "Custom Training Data set" described in subsection III-B2) time relatively insignificant, while still allowing accurately measuring the answering performance.

B. Identifying Training Seeds

Table 1 presents the results with various configurations of our general model. Since the goal of our general model is to provide training examples (seeds), we report "Precision at 10" averaged across the categories as our primary metric of interest. Since we only needed to check top 10, it was possible to manually inspect the results for all the test questions. For comparison, we also report our re-implementation of [23] and another baseline that used mutual information between the instance and the category from [18]. These are the only models that are capable of automatically providing training seeds for any arbitrary category without relying on curated ontologies, as described above in our literature review.

As Table 1 illustrates, the results are extremely encouraging and suggest the following: 1) The overall accuracy of the automatically provided training seeds is above 83%, which is higher than reported in the related work and is roughly equal to the human accuracy on the task. 2) Using Wikipedia as the only validation corpus is not sufficient.

TABLE II
TABLE 2 QUESTION ANSWERING ACCURACY FOR THE CONFIGURATIONS TESTED.

Model	Category Coverage	Answer Coverage	Answer Accuracy
Our Models:			
full configuration	100%	100%	**73%**
using only 7 seeds	100%	100%	71%
using only 5 seeds	100%	100%	65%
using 200 segments per candidate	100%	100%	69%
using 50 segments per candidate	100%	100%	59%
Baselines:			
no semantic category validation	0%	0%	10%
using only our general model	100%	100%	49%
Freebase	26%	95%	24%
NELL[8]	36%	50%	22%
MCG[9]	21%	74%	17%
our re-implementation of [36]	100%	100%	47%
perfect validation Oracle	100%	100%	83%
state of the art in similar settings from prior works			
[29][30][31][32][33][7]	100%	100%	40-70%

C. Question Answering Results

In this study, we focus on the approaches that do not rely on a manually pre-built Knowledge Base (KB), but check arbitrary categories and arbitrary candidates based on raw (unstructured) text only. We list such approaches as baselines in our Table 2 and describe in our literature review. Table 2 presents the question answering results for several configurations and various related baselines. "Answer Accuracy" is the proportion of correctly answered questions. The "Category coverage" column shows the percentage of the tested categories that were "recognized" by the approach. For those based on manual resources, such as Freebase, "Category coverage" is the proportion of the categories that are present in them as entities (exact match, ignoring upper/lower case). Similarly, "Answer Coverage" column shows the proportion of questions, the correct answers to which are contained in the resource. Again, since the variations of our approach are not relying on any ontologies and can evaluate any candidate-category pair on demand, their both Category and Answer coverage are reported as 100%. The numbers for NELL [8], MCG [9] and Freebase were estimated manually based on a random sub-sample of 100 questions from those that we used here. To estimate the "Answer Accuracy" for them, we counted the question answered correctly if and only if both the category and the correct answer are included in the ontology, thus assuming the reasoning mechanism will be able to match them to the question correctly. All the tested differences from the best value (in **bold**) are statistically significant at the level of 0.01. The QA system as reported in [36] used entire web as source of answers, taking them from the snippets provided by the search portals. Since we used a fixed corpus here, we simulated the search engine by the same TF-IDF ranking algorithm that we applied for our QA approach here.

The results in Table 2 support the following conclusions: 1) Our meta-learning-inspired two-tier model works better than a general-model alone, thus confirming the superiority of our 2-tier approach over prior works relying on a single model. 2) When combined with a simple PMI-based candidate answer scoring, our model delivers the state-of-the-art result on the types of questions tested. 3) The more data our model uses, the better its performance. 4) The coverage of the largest manually curated ontologies is not sufficient to match the state-of-the-art performance without involving additional reasoning mechanisms.

TABLE III
EXAMPLES OF **LOWEST** SCORING SEGMENTS FOR THE CATEGORY-SPECIFIC MODELS.

<seed> : <cat> life
<seed> and <cat> spirit guides
<seed> overview <cat> overview

TABLE IV
EXAMPLES OF **HIGHEST** SCORING SEGMENTS FOR THE CATEGORY-SPECIFIC MODELS.

segment	category
translation of <seed>	book
<seed> is the darkest <cat>	color
the <seed> is usually the tallest <cat>	player on a basketball team

Tables 3 and 4 present examples with lowest and highest scoring segments for the category-specific models. Normally, the lowest scoring segments are coming from noise such as accidental word combinations, advertising, html markup, etc. It is worth noticing, that in addition to learning positive indicators, our model learns to recognize noisy text segments and use them as negative evidence. The highest scoring segments consist of the expressions typical for instances of that category (e.g. *translation* for a *book*). Those segments are closely resembling validation patterns displayed by NELL online demo [8].

V. CONCLUSIONS, LIMITATIONS AND FUTURE WORK

We have successfully applied a two-tier approach inspired by meta-learning to the task of validating any (not trained apriori) category-instance pair, e.g. (*country, France), (movie, "Gladiator"), (Las Vegas hotel, "MGM Grand")*. No manual labeling is necessary for this beyond what was already involved in our study to create a general pattern-based model capable of providing high quality training seeds for the

RNN-based category specific models. Our approach can be considered complementary to those relying on large manually (or semi-automatically) assembled Knowledge Bases and approximate reasoning. Instead of pre-anticipating all the possible categories that may be needed to answer questions, the category verification is performed on-the-fly from raw text only, which is available in virtually all modern languages, even those that do not have such resources as e.g. Freebase[1] in English. Our approach can handle long and complex categories like *a city in southern Germany* as long as they exist in the validation corpus.

A number of limitations has been already discussed through our paper, and they can be addressed in future work, specifically: 1) Not relying on external search engines. A corpus larger than Wikipedia ill need to be used for that purpose, e.g. large crawled portions of the Web (e.g. Common Crawl) or social media posts. 2) Using machine learning approaches, e.g. RNN GRU-s to combine semantic verification and question answering into one framework. 3) Testing on larger sets of questions and categories. 4) Testing with other than question answering applications, e.g. automatically finding inconsistencies or missing data in manually created resources such as Freebase or Wikidata.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. E. Hinton, "Deep learning," in *Nature 521(7553)*, 2015.

[2] E. V. Clark, "First language acquisition," in *Cambridge University Press*, 2009.

[3] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, and D. Lin, "Knowledge base completion via search-based question answering," in *Pro-ceedings of the 23rd international conference on World wide web*, 2014.

[4] H. Miller, A. Fisch, J. Dodge, A. Karimi, A. Bordes, and J. Weston, "Key-value memory networks for directly reading documents," in *EMNLP*, 2016.

[5] D. A. Ferrucci, E. W. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. M. Prager, N. Schlaefer, and C. A. Welty, "Building watson: An overview of the deepqa project," in *AI Magazine 31*, 2010.

[6] E. Voorhees and L. P. Buckland, "Proceedings of the eleventh text retrieval conference," in *TREC*, 2005.

[7] W. Cui, H. Wang, Y. Song, S. W. Hwang, and W. Wang, "Kbqa: learning question answering over qa corpora and knowledge bases," in *VLDB*, 2017.

[8] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Better-idge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krish-namurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling, "Never-ending learning," in *Conference on Artificial Intelligence (AAAI)*, 2015.

[9] Z. Wang, H. Wang, J.-R. Wen, and Y. Xiao, "An inference approach to basic level of categorization," in *ACM International Conference on Information and Knowledge Management (CIKM)*, 2015.

[10] R. Calandra, F. Hutter, H. Larochelle, and S. Levine, "Workshop on meta-learning," in *NIPS*, 2017.

[11] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *in Proceedings of the 14th International Con-ference on Computational Linguistics*, 1992.

[12] S. Brin, "Extracting patterns and relations from the world wide web," in *Proceedings of the WebDB Workshop at EDBT*, 1998.

[13] M. Poesio, T. Ishikawa, S. S. im Walde, and R. Viera, "Acquiring lexical knowledge for anaphora resolution," in *3rd Conference on Language Resources and Evaluation (LREC)*, 2002.

[14] K. Ahmad, M. Tariq, B. Vrusias, and C. Handy, "Corpus-based thesaurus construction for image retrieval in specialist domains," in *Proceedings of the 25th European Conference on Advances in Information Retrieval (ECIR)*, 2003.

[15] R. Girju and M. Moldovan, "Text mining for causal relations," in *In Proceedings of the FLAIRS Conference*, 2002.

[16] K. K. Markert, N. Modjeska, and M. Nissim, "Using the web for nominal anaphora resolution," in *EACL Workshop on the Computational Treatment of Anaphora*, 2003.

[17] P. Cimiano, G. Ladwig, and S. Staab, "Gimme' the context: Context-driven automatic semantic annotation with c-pankow," in *Proceedings of the 14th World Wide Web Conference*, 2005.

[18] A. Yates and O. Etzioni, "Unsupervised methods for determining object and relation synonyms on the web," in *Journal of Artificial Intelligence Research 34*, 2009.

[19] S. Schlobach, M. Olsthoorn, and M. de Rijke, "Type checking in open-domain question answering," in *BNAIC*, 2004.

[20] S. P. Igo and E. Riloff, "Lexicon induction with web-based corroboration," in *Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*, 2009.

[21] M. Fleischman and E. Hovy, "Fine grained classification of named entities," in *In Proceedings of the Conference on Computational Linguistics*, 2002.

[22] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *25th international conference on Machine learning*, 2008.

[23] D. Roussinov and O. Turetken, "Semantic verification in an online fact seeking environment," in *ACM Conference on Information and Knowledge Management*, 2007.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.

[25] O. Levy, S. Remus, C. Biemann, and I. Dagan, "Do supervised distributional methods really learn lexical inference relations?" in *NAACL*, 2015.

[26] V. Shwartz, Y. Goldberg, and I. Dagan, "Improving hypernymy detection with an integrated path-based and distributional method," in *Proceedings of Association of Computational Linguistic*, 2016.

[27] S. Gupta and C. D. Manning, "Improved pattern learning for bootstrapped entity extraction," in *CoNLL*, 2014.

[28] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100, 000+ questions for machine comprehension of text," in *EMNLP*, 2016.

[29] P. McNamee, R. Snow, P. Schone, and J. Mayfield, "Learning named entity hyponyms for question answering," in *IJCNLP*, 2008.

[30] A. Bordes, N. Usunier, S. Chopra, and J. Weston, "Large-scale simple question answering with memory network," in *ICML*, 2015.

[31] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading wikipedia to answer open-domain questions," in *55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[32] H. Sun, H. Ma, W. Yih, C. T. Tsai, J. Liu, and M. W. Chang, "Open domain question answering via semantic enrichment," in *Proceedings of the 24th International Conference on World Wide Web*, 2015.

[33] C.-T. Tsai, W.-T. Yih, and C. J. Burges, "Askbing: A web-based question answering system," in *Technical report, Microsoft Research*, 2014.

[34] J. Liang, Y. Xiao, H. Wang, Y. Zhang, and W. Wang, "Probase+: Inferring missing links in conceptual taxonomies," in *TKDE*, 2017.

[35] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.

[36] D. Roussinov, "Towards semantic category verification with arbitrary precision," in *ICTIR*, 2011.

[37] C. Manning and H. Schutze, in *Foundations of Statistical NLP*, 2000.

[38] K. Wang, C. Thrasher, E. Viegas, X. Li, and B.-j. P. Hsu, "An overview of microsoft web n-gram corpus and applications," in *Proceedings of the NAACL HLT 2010 Demonstration Session*, 2010.

[39] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[40] P. Baudis and J. Sedivy, "Modeling of the question answering task in the yodaqa system," in *International Conference of the CrossLanguage Evaluation Forum for European Languages*, 2015.

[1]Now discontinued by Google