Bootstrapping a Game with a Purpose for Common Sense Collection

AMAÇ HERDAĞDELEN[1] and MARCO BARONI, CIMeC, University of Trento

Text mining has been very successful in extracting huge amounts of commonsense knowledge from data, but the extracted knowledge tends to be extremely noisy. Manual construction of knowledge repositories, on the other hand, tends to produce high quality data in very small amounts. We propose an architecture to combine the best of both worlds: A game with a purpose that induces humans to clean up data automatically extracted by text mining. First, a text miner trained on a set of known commonsense facts harvests many more candidate facts from corpora. Then, a simple slot-machine-with-a-purpose game presents these candidate facts to the players for verification by playing. As a result, a new dataset of high precision commonsense knowledge is created. This combined architecture is able to produce significantly better commonsense facts than the state-of-the-art text miner alone. Furthermore, we report that bootstrapping (i.e., training the text miner on the output of the game) improves the subsequent performance of the text miner.

Categories and Subject Descriptors: I.2.4 [**Artificial Intelligence**]: Knowledge representation—*Semantic Networks*; J.4 [**Computer Applications**]: Social and Behavioral Sciences; H.1.2 [**Models and Principles**]: User/Machine Systems—*Human Factors*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms: Algorithms, Design, Experimentation, Human factors

Additional Key Words and Phrases: games with a purpose, common sense, knowledge extraction, natural language processing, facebook

1. INTRODUCTION

Everyday knowledge, otherwise known as *common sense*, is a vast network of generic facts which everyone assumes that – almost – everyone else knows and thus enables people to function in diverse conditions of daily life and coordinate their behaviors. Such knowledge looks naïve and superficial at first look (bedrooms have floors, grocery has a price, ...) but essentially it is what sets apart human beings from the state-of-the-art artificial intelligence systems [Minsky 2000]. The importance of representing everyday knowledge in a computational system in order to attain human-level intelligence has been acknowledged ever since the early days of artificial intelligence research [McCarthy 1959] and the area continues to be a hot topic for recent AI research.

The attacks at the commonsense knowledge problem have ranged from manual creation of commonsense knowledge repositories, such as the Cyc database, to knowledge-poor induction of such knowledge from the text available on the Web [Lenat 1995; Banko et al. 2007]. However, the manual method is laborious and expensive while the text mining methods are prone to noise. Banko et al. estimate that about 20% of the millions of generic facts they extracted from the Web with a state-of-the-art large scale information extraction system are wrong [Banko et al. 2007].

A more human-computation-centered approach to collecting commonsense knowledge is to recruit laypeople from the Web and have them contribute to a knowledge base. The Open Mind Common Sense project [Speer 2007] relies on the good will of Web surfing volunteers and has been quite successful at collecting tens of thousands of generic facts from ordinary people. An alterna-

[1] Author's current affiliation is New England Complex Systems Institute (NECSI).

Authors' e-mail addresses: amac@herdagdelen.com and marco.baroni@unitn.it.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

tive to volunteer work is that of *games with a purpose* [Von Ahn 2006], inducing Web surfers to contribute various kinds of useful knowledge while playing and having fun. Recently, social networking sites like Facebook have been used to deploy such games for easier access to a large user base [Rafelsberger and Scharl 2009].

In this study, we present a combination of a text mining algorithm and a human computation framework that allows us to attain high throughput (in terms of annotated assertions) while keeping the precision at acceptable levels in commonsense knowledge harvesting. The first part of the proposed architecture is BagPack (**Bag**-of-words representation of **Pa**ired **c**oncept **k**nowledge), an algorithm based on a vector-space model for representing commonsense assertions (or, more generally, statements about the semantic relation linking two concepts). For a given relation (e.g. LocationOf, MotivatedByGoal, ...), it evaluates a set of assertions and outputs a list of candidates ranked according to their likelihood of being true. The output of BagPack is the input of the second part of our proposed architecture, Concept Game, which is a Facebook slot-machine-like game that lets players validate these candidates. The players are shown many "random" assertions and are asked to identify those that make sense in order to gain points.

The abundance of the candidate assertions mined by BagPack allows us to free the players from the burden of producing the commonsense facts. All they have to do is to express their assent (or lack of it). We leverage this opportunity by implementing a fast-paced game which does not place a high cognitive load on the player. The candidate assertions extracted from corpora also allows us to employ a more data-driven approach to extend the current knowledge bases; unlike in other human-computation approaches, we can tap the corpora to collect assertions that volunteers/players do not typically provide. In addition, the slot-machine game provides a convenient excuse for the noise in the displayed candidates. Concept Game is a game of chance and a player is expected to see many meaningless assertions before "hitting the jackpot". Thus, the low precision of the text miner module becomes a natural part of the game experience, not a source of frustration. Another advantage is that, besides their inherent value, such cleaned-up data can be used to assess the quality of text mining and fed back to the algorithm as labeled training materials.

The main goal of the current paper is to introduce our combined architecture, and to show that not only it allows us to collect significantly better commonsense facts than the state-of-the-art text miner alone but also that the commonsense dataset it constructs can be used to improve text mining performance in subsequent runs. After reviewing some related work in Section 2, we describe our text mining algorithm in Section 3. We provide preliminary results both on a widely used semantic task where we can compare its performance with the state of the art, and tuned to extract commonsense assertions in Section 4. In Section 5, we describe the design of our game. In Section 6, we describe our combined mining+gaming experimental procedures and report the results. Section 7 concludes the paper with main achievements and future directions.

2. RELATED WORK

To the best of our knowledge, our proposal is the first system integrating commonsense harvesting by text mining and a game with a purpose. However, examples of both approaches – implemented individually – exist and we will review the text mining and human computation approaches separately.

2.1. Repositories of Common Sense

Common sense consists of assumptions and known facts about daily life shared by a great majority of people. The ontologist Barry Smith stresses the massive amount of common sense: "Common sense includes a massive storehouse of factual knowledge about colours and sounds, about time and space, about what foods are edible and what animals are dangerous." [Smith 1995]. For an average citizen of the modern world, we can easily extend this definition to cover facts like "One should turn off his cellphone while attending a concert." or "People eat breakfast in the morning." [Lieberman 2008]. It looks tricky to appeal to majority for a definition ("what exactly is a great majority of people?") but this appeal is the very essence of the functional benefits of common sense. It allows

us to communicate with other people without being verbose to the point that communication is impossible. For an efficient communication, we can rely on the set of beliefs and facts we share with our interlocutor [Wright et al. 1996].

Representation of commonsense knowledge is at the core of AI research and there have been several attempts to solve this problem. One of the well-known AI research projects that attempt to solve the commonsense problem is the Cyc project initiated by Douglas Lenat in 1984 [Lenat 1995]. Today, Cyc is a commercial knowledge base (owned and developed by Cyccorp Inc.[2]) that intends to capture a significant portion of commonsense knowledge. It employs a detailed ontology of concepts, actions and rules and has its own special formal language to enter new facts. The underlying semantics extends beyond a first-order predicate logic to allow an expressive representation. ResearchCyc is a limited version of Cyc available under a free license and – as of 2005 – it was containing approximately a million assertions about more than a hundred thousand symbols [Ramachandran et al. 2005]. The fact that Cyc is a formal repository of commonsense knowledge increases the amount of effort to construct, extend, and maintain it because one needs to know about the formal semantics and syntax underlying the knowledge base.

Open Mind Common Sense (OMCS) is another commonsense database project initiated by MIT's Media Lab in 1999 [Speer 2007]. A key difference between OMCS and Cyc is that the former relies on semi-structured snippets of natural language phrases to represent commonsense knowledge instead of a logical formalism. For example, an instantiation of the commonsense relation of AtLocation between the two concepts ashtray and bar can be represented as "Something you find at a bar is an ashtray". Here, the template "Something you find at a {} is an {}." corresponds to LocationOf and the concepts are normalized forms of natural language phrases (e.g., "ashtrays", "an ashtray", and similar phrases are normalized to ashtray by means of lemmatization and stop-word removal.

OMCS's choice of semi-structured phrases reduces the deductive inference power of the system because of the ambiguities introduced by natural language and the imperfect mapping of phrases to concepts. On the other hand, the less formal representation allows to recruit laypeople to enter facts into OMCS because almost no training is required to formulate facts. In fact, the entire content of OMCS is based on the volunteers' efforts. Since 1999, more than 16,000 people contributed to OMCS via a web-based interface, resulting in more than 700,000 English facts [Havasi et al. 2009]. Moreover, the ambiguity and redundancy introduced by the natural language representation can be an advantage for flexible inference. Interested readers can refer to Liu and Singh [Liu and Singh 2004] for a detailed discussion of the advantages and disadvantages of the natural language representation.

ConceptNet[3] is a semantic network that is based on the facts contained in OMCS [Liu and Singh 2004]. The vertices are the concepts and the links come from a closed set of common sense relations such as AtLocation, CapableOf, and HasLastSubEvent. We rely on ConceptNet to seed our text miner (see Section 4.1 below).

2.2. Mining for Common Sense

Corpus-based techniques proved themselves reliable methods of knowledge extraction by using statistical analysis of frequently occurring patterns in large amounts of text. They are widely adapted in domains ranging from machine translation to biomedical text mining [Buitelaar and Cimiano 2008].

Can we follow a corpus-based path for our own problem and mine common sense from corpora? At a first look, the answer should be no because, by definition, commonsense facts go unstated explicitly in discourses. A robber does not start his threat by reminding his victim that "guns shoot bullets" and "bullets kill people". Rather, he simply says "Give me your wallet or I will shoot!" (assuming, for the sake of argument, that we have a corpus of robberies). As a matter of fact,

[2] http://www.cyccorp.com/
[3] http://conceptnet.media.mit.edu/

precisely this observation motivates the effort for constructing commonsense knowledge bases – if common sense was readily found in text (context in general) no one would have to collect it!

Nonetheless, whether it is possible to extract general – and of course true – facts about the daily life from corpora is an empirical question and the literature provides encouraging results on this problem. A key observation is that many types of commonsense knowledge are reflected at the surface level in text, albeit somewhat indirectly, as part of statements whose communicative aim is not to state a commonsense truth. Lucy Vanderwende (2005) cites the following example [Vanderwende 2005]:

> A bat is the only mammal that can truly fly.

Upon reading this sentence, even if one knows nothing about what a mammal or bat is, one can deduce that

— bats can fly
— mammals (mostly) do not fly.

Therefore, algorithms that leverage such implicit clues can be employed to extract commonsense knowledge. Vanderwende herself presents a proof of concept to show how such a system can be implemented.

In another related study, Strohmaier and Kröll (2009) analyze the search query logs released by commercial search engines and conclude that "search query logs are a potential source of common human goals." [Strohmaier and Kröll 2009]. Their methodology is to compare the verb phrases extracted from query logs to the verb phrases already contained in ConceptNet. A significant amount of overlap between the goals extracted from corpus and the goals contained in ConceptNet motivates their conclusion.

The aforementioned two studies tell us how and why corpus-based approaches can succeed in the task of commonsense knowledge extraction. The following studies are examples of some systems that attempt to carry out this task.

KNEXT (KNowledge EXtraction from Text) is a system proposed for extracting "general world knowledge from miscellaneous texts, including fiction" [Schubert and Tong 2003]. Schubert and Thong (2003) provide an extensive evaluation of the output of KNEXT on the British National Corpus (a balanced and representative collection of spoken and written English samples, containing 100 million words)[4] and according to their evaluation based on five human judges, almost 60% of the generated propositions are found to be "reasonable general claims" by any given judge. However, the agreement between judges on individual facts is not extremely high. Considering only cases where all five judges agree, the ratio of "reasonable general claims" reduces to one third. Recently, it was also shown by Schubert and collaborators that – with significant post-processing and filtering of output – more noisy corpora such as texts coming from blogs can be used as another source of commonsense knowledge [Gordon et al. 2010].

Using web-based text as a source for commonsense knowledge is indeed becoming a popular technique. The abundance of text coming from blogs, web pages, discussions in newsgroups and other social media (e.g. Twitter, Facebook) allows us to cover a wide array of domains and extract a great number of facts. The sheer amount of input also helps to fight with the reported noise in the output of knowledge extraction systems. It is possible to apply very strict filtering conditions and focus on a – relatively – small subset of the output to obtain high quality facts with a trade-off in recall (the amount of extracted knowledge relative to the amount that could be extracted in ideal conditions). Below we discuss some examples of this approach.

In 2005, a research group from Cyccorp reported a preliminary experiment for extending the knowledge base of Cyc by issuing template-based queries to a commercial search engine and analyzing the results [Matuszek et al. 2005]. The first step in their approach is to identify missing pieces of knowledge in Cyc such as the missing information on the founder of the Palestine Islamic Jihad

[4] http://www.natcorp.ox.ac.uk/

Organization, represented as a tuple (foundingAgent PalestineIslamicJihad ?WHO). In subsequent steps, they reformulate the missing information as a natural language template such as "PIJ, founded by *", issue a search to Google by using this template as the query, and analyze the resulting snippets. An example snippet returned by the search engine is "PIJ founder Bashir Musa Mohammed Nafi is still at large..." which suggests that Bashir Musa Mohammed Nafi is a likely candidate to be inserted into the initial incomplete tuple. After some post-processing and filtering – which includes consistency checking with the already known facts in Cyc – the final candidates are presented to a human volunteer or an expert and those that pass the final evaluation are inserted in Cyc.

The ConceptMiner of Ian Eslick [Eslick 2006] is another system that populates commonsense assertions by issuing pattern-based queries to Google. The system is based on ConceptNet (the semantic network that is based on OMCS) and the aim is to extend ConceptNet by finding new pairs of concepts that are related to each other by one of the commonsense relations contained in ConceptNet. The key idea in ConceptMiner is similar to the one discussed in Vanderwende (2005); a set of known facts contained in ConceptNet serves as a seed and ConceptMiner extracts typical surface-level linguistic expressions that bind pairs of concepts bound by a given relation. For instance, for a pair of concepts dog and bark which are related to each other by CapableOf, a search operation can result in the following phrases: "when a dog barks", "the dog never stopped barking", etc...[5] Then, another search session is used to find further instantiations of those surface forms to get new candidates of concept pairs for the relation (e.g. the queries "when a {} {}" or "the {} never stopped {}" are issued and the results are parsed to extract new pairs of concepts). Statistical analysis and extensive filtering of the candidates result in a substantially smaller but high quality set of assertions that can be inserted in ConceptNet.

Another application which uses ConceptNet as a seed and attempts to extend it by using the web as a corpus is that of Yu and Chen presented in 2010 [Yu and Chen 2010]. In this approach, for each relation in question, a dataset of assertions is constructed from the facts stored in ConceptNet. In addition to the original ConceptNet assertions (which serve as positive training instances), an equal number of randomly-crafted assertions are added to the datasets to serve as negative instances. Then, each instance (i.e., each pair of concepts) in a training set is represented in a vector space that captures the co-occurrence patterns in a corpus (Yu and Chen use Google Web 1T 5-Gram corpus). The results show that a support vector machine (SVM) trained on a subset of the initial dataset is able to discriminate meaningful facts from random facts in the left-out part of the dataset with an accuracy that is significantly higher than random performance. The accuracy of the trained SVMs range from 55% to 85% for different relations and model parameters (where 50% is the random baseline).

Never-Ending Language Learner (NELL) is a system that was recently implemented as part of the *Read the Web* project to extract structured facts from documents found in the Web [Carlson et al. 2010]. NELL aims to populate and grow a knowledge base that is shared between several learner components (e.g., a wrapper-based learner that extracts lists from pages, a rule learner, a pattern learner). The knowledge base is structured according to an ontology specifying a set of semantic categories (e.g., *city*, *company*) and relations between instances of these categories (e.g., *hasOfficesIn(organization, location)*). Initially, the knowledge base is populated with a small set of instances of the categories and the relations. NELL analyzes the Web documents and populates even more instances based on the output of its components. Sharing the same knowledge base allows the components to be trained in a coupled setting because they can learn from each other's output. NELL works in an incremental way and adds only a few instances that are likely to be true at each step. Using independent components may help to avoid semantic drift (i.e., accumulation of errors leading to deteriorating performance in the long run), but still a human intervention is deemed essential in maintaining the overall quality of NELL's output.

[5]This is a very simplified example. In the real setting, ConceptMiner applies other analyses like POS-tagging and lemmatization.

2.3. Human Computation for Common Sense

Quinn and Bederson classify different approaches to harnessing the computation capabilities of humans [Quinn and Bederson 2009]. Among the several dimensions they use for categorization, the following are relevant for our discussion: Motivation of users, techniques for coping with noise, minimum participation time, and cognitive load placed on the player. Below, we discuss the human-computation-based approaches to commonsense knowledge collection that we are aware of along these dimensions.

Verbosity [Von Ahn et al. 2006] is a word-guessing game very much like the commercial game "Taboo". In a single round of the game, a describer tries to tell the secret word to a guesser by filling in one or more of the slot-based templates he is given. The templates are crafted in order to collect commonsense knowledge about the secret word. Some examples are "It is a type of ___", "About the same size as ___". If the guesser is able to find out the secret word, the clues that the describer provided are considered to be true and stored. Common Consensus [Lieberman et al. 2007] is another game with a purpose based on the popular TV show "Family Feud" where the player is asked to guess the popular answers for a given question like "Something that is likely to be found in a kitchen is ___". Both games's output is used to populate facts in OMCS and consequently in ConceptNet [Speer et al. 2010].

Both Verbosity and Common Consensus are games with a purpose where the players' primary motivation is having fun and commonsense knowledge collection is a side effect of the game playing. Thus, they differ from the volunteer-based knowledge collection effort of OMCS. In order to cope with noisy input, both games can employ redundancy and accept a fact as true only after it is asserted by more than one player. However, the players are free to introduce the facts they like and that means many valid facts may be asserted only once – resulting in a sparse dataset. The games can only guide the players to provide information about a given domain or a target concept but they lack the ability to validate a given commonsense fact. Coupled with the tendency of players to abuse the system, especially for Verbosity where the players try to do their "best" to convey the secret word, noise reduction and validation call for non-trivial and usually heuristics-based approaches [Speer et al. 2010]. For both games, the overhead of playing is quite low and players can almost immediately start to contribute. However, the burden of producing commonsense facts are placed on the players. In Verbosity, it is up to the describer to think and find good clues and in Common Consensus, the player has to come up with plausible candidates for the popular answers. Thus, the games are cognitively engaging and the cognitive load on the player is quite high.

Another application that is related to Cyc is the game called FACTory (`http://game.cyc.com/`). In FACtory, the player is shown a set of commonsense statements and is asked to express his opinion about the statement. The FACTory's commonsense statements are generated from the CYC repository, and players must tell whether they think the statements are true or false. Extra points are awarded when a player agrees with the majority answer for a fact and a certain consensus threshold has been reached. Although the system is presented as a game – and thus, the motivation of answering the questions is expected to be having fun – in its current stage, there is no clear fun aspect in FACTory, that looks like an interactive and marginally less boring mechanism to collect human opinions about a given set of commonsense statements. An important difference between FACTory and the previous two games is that FACTory relies on Cyc as a source of commonsense statements. Therefore, all it has to do is to ask for the opinion of the player. This has the benefit of reducing the cognitive load on the player (it is easier to say yes to the question "Can salmon be found in a fridge?" rather than to try to come up with "salmon" as an answer to the question "What can be found in a fridge?".

There are other attempts to crowdsource commonsense data evaluation/collection, by utilizing services like Amazon's Mechanical Turk (`http://www.mturk.com/`), e.g., Gordon et al. [2010]. We consider these payment-based crowdsourcing methods as complementing the games-with-a-purpose approach rather than competing with it. For the evaluation of small datasets, crowdsourcing may be a convenient alternative, but as the amount of data to be annotated increases so

does the cost of annotation. In contrast, the operational costs of games are usually almost constant (e.g., a small monthly reward to motivate players) and enlarging the user base would not incur any additional costs. Current estimates are that the unit cost per response for our Concept Game in its initial phase has been comparable to, if not less than, the cost reported for Mechanical Turk services (around 300 to 500 responses per USD spent) [Gordon et al. 2010]. As (read if) the game becomes more popular among Facebook users, the unit cost will get much lower.

3. BAGPACK

The approach to the extraction of semantic information taken by our text miner of choice, namely BagPack (**Bag**-of-words representation of **Pa**ired **c**oncept **k**nowledge), is to construct a vector-based representation of a pair of terms in such a way that the vector represents both the contexts where the two terms co-occur (that should be very informative about their relation) and the contexts where the single terms occur on their own (possibly less directly informative but also less sparse). As far as we know, BagPack is the first semantic mining algorithm that exploits together information about the contexts in which paired terms co-occur, as well as contexts in which the pair components occur separately.

For a given pair of concepts, BagPack constructs three different sub-vectors, one for the first term (recording frequency of sentence-internal co-occurrence of the first term with context items which may be unigrams – i.e., single words – or n-grams – i.e., word sequences – depending on implementation), one for the second (with the same kind of information), and one for the co-occurring pair (keeping track of the items that occur in sentences where both terms occur). The concatenation of these three sub-vectors is the final vector that represents the pair. In Herdağdelen and Baroni [2009], where we first introduced it, the vector space that BagPack constructs was employed exclusively in a supervised setting: The vectors constructed for labeled pairs (including positive and negative examples of the target relation) were fed to a support vector machine that was then used to classify or to rank unlabeled pairs according to their confidence scores. In this study we employ BagPack also in an unsupervised setting where we use the cosine similarities between the vectors directly to assess semantic similarity.

Which specific text mining algorithm is employed before the human computation step is not a crucial aspect of our combined architecture of text mining and human computation. We would like to think of the text miner as a "plug-in" and any reasonable model which produces a list of candidate assertions ranked according to their likelihood of being meaningful would do the job. That said, we use BagPack because it is a model that was readily available to us, which proved itself as a robust and reliable representation of concept pairs. A detailed description of BagPack is beyond the scope of this paper; therefore, we limit our discussion of BagPack in this study to its comparison with other algorithms on two semantic benchmarks: The first experiment is based on the SAT analogy recognition task. The second requires the classification of meaningful and meaningless assertions expressing three ConceptNet relations.

For both tasks, the same BagPack implementation and source corpora are used as described below.

We carried out our tests on the Web-derived English Wikipedia and ukWaC corpora, about 2.8 billion tokens in total (we use the pre-processed versions from `http://wacky.sslmit.unibo.it`). We did not carry out a search for "good" parameter values. Instead, the model parameters are generally picked at convenience to ease memory requirements and computational efficiency. Once we construct the vectors for a set of word pairs, we get a *co-occurrence matrix* with pairs on the rows and the features (words that co-occur with the pairs) on the columns (including pair- and single-occurrence features). PMI feature weighting is applied to the co-occurrence matrix [Church and Hanks 1990].

Please refer to the more thorough description and evaluation of BagPack we report in Herdağdelen and Baroni [2009], that also shows how BagPack is a highly adaptive algorithm, reaching (near) state-of-the-art performance on a variety of tasks.

Table I. Percentage accuracy in solving SAT analogies with 95% binomial confidence intervals. Model sources: [1][Turney and Littman 2005]; [2][Turney 2006b]; [3][Turney 2006a]; [4][Turney 2008]; [5][Biciçi and Yuret 2006]; [6][Baroni and Lenci 2010]; [7][Quesada et al. 2004]

model	*accuracy*	*95% CI*	*model*	*accuracy*	*95% CI*
Human[1]	57.0	52.0-62.3	LSA[7]	42.0	37.2-47.4
LRA-06[2]	56.1	51.0-61.2	**BagPack**	39.6	34.6-44.7
PERT[3]	53.3	48.5-58.9	LRA-10[6]	37.8	32.8-42.8
PairClass[4]	52.1	46.9-57.3	PMI-IR-06[2]	35.0	30.2-40.1
VSM[1]	47.1	42.2-52.5	DepDM[6]	31.4	26.6-36.2
k-means[5]	44.0	39.0-49.3	LexDM[6]	29.3	24.8-34.3
TypeDM[6]	42.4	37.4-47.7	Random	20.0	16.1-24.5

3.1. SAT Analogy Recognition Task

The SAT analogy questions task was introduced by Turney et al. [2003]. In this task, there are 374 multiple choice questions with a pair of related words as the *stem* (e.g., *wallet-money*) and 5 other pairs as the *choices* (e.g., *safe-lock*, *suitcase-clothing*, *camera-film*, *setting-jewel*, *car-engine*). The correct answer is the choice pair which has the relationship most similar to that in the stem pair (*suitcase-clothing* in this example). We chose to evaluate BagPack on this task because it is a standard benchmark for corpus-based semantic models, and it thus allows us to assess the relative performance of BagPack with respect to many state-of-the-art models. Moreover, the sort of analogical reasoning required by the task (you use a wallet to store money like you'd use a suitcase to store clothing) is a classic example of reasoning with commonsense knowledge, and we might hope that good performance of a model on the SAT cues good performance on commonsense tasks in general.

3.1.1. Task-specific setup. We adopt an unsupervised approach to answer the SAT questions. The cosine similarities between the choices and the stem pair are computed for each question and the choice which is the most similar to the stem is picked as the predicted answer.

3.1.2. Results. Table I compares the performance of BagPack to that of recent studies in which various state-of-the-art text miners have been tested on the SAT challenge[6]. Overall, the performance of BagPack is not quite at the top of state of the art, but comparable to that of several recent systems, and in particular to those implemented by Baroni and Lenci [2010], that used a similar, slightly larger corpus (ukWaC, Wikipedia plus the 100M word British National Corpus).

3.2. Classifying ConceptNet relations

We perform a more direct comparative evaluation of BagPack as a commonsense miner on the task of deciding whether an assertion from the ConceptNet knowledge base instantiates a certain commonsense relation or not. We compare BagPack with two recent text miners that are readily available to us. Latent Relational Analysis (LRA) was originally proposed by Turney [2006b], and it has been shown by its author to reach state-of-the-art performance in a number of different tasks in the original paper and subsequent work.

LRA is similar to BagPack in that it harvests lexical patterns that connect two concepts in context, but, unlike BagPack, it does not rely on contextual information relating to the two concepts separately. It tackles data sparseness by harvesting patterns also for pairs of concepts similar to the target ones (e.g., using *automobile-wheel* as a proxy for *car-wheel*), and by applying the Singular Value Decomposition to the co-occurrence matrix representing the target pairs. Type-based Distributional Memory (TypeDM) was among the best models across a range of semantic tasks in the recent extensive evaluation of Baroni and Lenci [2010]. TypeDM exploits lexical patterns as well as syntactic dependency relations to attain a richer representation of the contexts in which pairs co-occur. Moreover, it tries to measure the *variety* of contexts in which the pairs co-occur, rather

[6]See `http://aclweb.org/aclwiki/` for further information and references

Table II. Summary of the commonsense datasets used in this paper. "Labeling" stands for the annotation of assertions as meaningful or meaningless. The two datasets, Wikipedia candidates and Bootstrap, contain the same assertions but with different labels based on different labeling procedures.

Dataset	Section	Source	Labeling based on
3-relation ConceptNet	3.2	ConceptNet	ConceptNet confidence scores
5-relation ConceptNet	4.1	ConceptNet	Expert raters
Wikipedia Candidates	4.3	Wikipedia	Expert raters
Bootstrap	6.1/6.2	Wikipedia	Concept Game players
Evaluation	6.2	Wikipedia	Concept Game players

than simple frequency of co-occurrence. It uses a form of smoothing-by-similar-pairs analogous to the one of LRA.

For TypeDM, we exploited the precompiled version available from `http://clic.cimec.unitn.it/dm`. For LRA, we adopt the implementation described in Baroni and Lenci [2010], accepting all their parameter specifications. For both methods, the parameter choices we are inheriting have been shown to be effective across a number of tasks by Baroni and Lenci [2010]. The algorithms are trained on the same corpus used for BagPack, enriched with the 100M tokens of the British National Corpus. Both algorithms, like BagPack, produce a co-occurrence matrix representing pairs of words in terms of contextual information, that we then feed as feature vectors to the same supervised classifier, for full comparability.

3.2.1. Task-specific setup.

Data. We focus on three of the five ConceptNet relations we will work with below when we test our combined text miner + game architecture, namely IsA, AtLocation and HasProperty (see Section 4.1 below for examples and discussion). For the remaining two relations (MotivatedByGoal and SymbolOf), the TypeDM model has very low coverage, and we do not want differences in coverage to trivially impact comparative performance. Also because of coverage issues, we restricted the choice of assertions from ConceptNet to those that do not involve concepts expressed by more than one word (TypeDM only contains single-word concepts). We remark that, independently of performance, the possibility of harvesting assertions containing multi-word concepts is a big advantage of BagPack over many related algorithms. Approximately 65% of the assertions represented in the five ConceptNet relations discussed in this paper contain at least one multi-word expression.

For each relation, we randomly picked from ConceptNet 4 250 assertions that exemplify the relation and 250 assertions that, somehow, were introduced to ConceptNet by a user as instantiating the relation, but later demoted by other users as being meaningless. (for HasProperty we could only extract 244 assertions of this kind). The latter were used as challenging negative examples since they are not totally random bindings of a relation and two concepts, but at one point they were conceived as a likely candidate of being true by at least one person.

Tuning the models. As far as BagPack is concerned, for each pair represented in the dataset, a vector was constructed in the same way as it was done for the SAT task. The TypeDM vectors were extracted from the precompiled TypeDM resource, and LRA vectors for the pairs were constructed following the Baroni and Lenci procedure [Baroni and Lenci 2010]. The following steps are identical for all models. For each relation we trained a SVM and tested its performance on a test set in a 10-fold cross-validation setting which was independently repeated 10 times itself. The results are based on the averages over all folds. Following the suggestion of Hsu and Chang (2003), before SVM training, each feature t's $[\hat{\mu}_t - 2\hat{\sigma}_t, \hat{\mu}_t + 2\hat{\sigma}_t]$ interval is scaled to $[0, 1]$, trimming the exceeding values from upper and lower bounds (the symbols $\hat{\mu}_t$ and $\hat{\sigma}_t$ denote the average and standard deviation of the feature values respectively). We use the C-SVM classifier as implemented in the Matlab toolbox of Canu et al. [2005] with a linear kernel and the cost parameter C set to 1.

3.2.2. Results. As performance measure of the models, we use the area under the ROC curve (AUC). This measure can be interpreted as the probability that a classifier will rank a random pos-

Table III. Comparison of BagPack, DM, and LRA on three ConceptNet relations. Numbers in brackets are 95% confidence intervals. For IsA and HasProperty, the differences between the mean AUC of BagPack and the other two methods are found to be statistically significant according to a two-tailed t-test ($p < 0.05$).

Relation	BagPack	DM	LRA
AtLocation	0.63 [0.62-0.65]	0.65 [0.63-0.67]	0.62 [0.61-0.64]
IsA	0.87 [0.86 - 0.88]	0.78 [0.77-0.79]	0.80 [0.79-0.81]
HasProperty	0.75 [0.74-0.77]	0.66 [0.65-0.67]	0.64 [0.63-0.66]

itive instance higher than a random negative instance [Fawcett 2006]. An AUC value of 0.5 means chance performance. Traditional measures such as precision and recall are not suitable in our task for several reasons including the following. 1) Both precision and recall need a threshold value applied on the posterior probabilities of the models, and deciding on a threshold is not a trivial task. For a fair and meaningful comparison of different models we would need to carry out extensive validation experiments, and that is beyond the scope of this paper. 2) AUC is invariant to relative class distributions [Airola et al. 2011]. Considering that we do not have a reliable estimate on the prior class probabilities in general (i.e. the ratio of meaningful assertions among all possible candidate assertions that can be constructed by using a corpus), AUC allows us to focus on the discriminative power of a model without having to pick a threshold and without worrying about the class distributions of a particular dataset.

The average AUC values for the three relations are given in Table III. For two of the three relations, IsA and HasProperty, BagPack performs significantly better than DM and LRA. For AtLocation, we did not observe any significant difference between the performances of the models.

4. USING BAGPACK TO MINE COMMONSENSE ASSERTIONS

Having evaluated BagPack on a generic semantic mining task (SAT) and compared its performance in recognizing commonsense assertions represented in ConceptNet, we now turn our attention to its application to commonsense mining, with the first of a series of experiments in which we attempt to extract new commonsense assertions of the sort that are stored in ConceptNet from the Web (note the difference with the classification task in Section 3.2, where the test set is also derived from CN, and thus no new assertions are mined from free Web text). A summary of the datasets that we use in this study is presented in Table II.

The training examples fed to BagPack come from the ConceptNet database, whereas evaluation is carried on a candidate set of assertions mined from Wikipedia. In this task, we use a support vector machine (SVM) that is trained on labeled examples of the *training set* (i.e., ConceptNet-based assertions) and the confidence scores of the SVM on the *test set* (i.e., Wikipedia-based candidate assertions) are used to rank and evaluate the performance.

4.1. Training materials

The initial training datasets are based on the assertions contained in ConceptNet 4 which is a freely available semantic network associating pairs of concepts with more than 25 semantic relations. In our study, we focus on five relations that represent rather different ways in which concepts are connected and correspond to more (IsA) or less (SymbolOf) traditional ontological relations. The relations tend, moreover, to link words/phrases from different syntactic classes: IsA (*cake, dessert*); AtLocation (*cake, oven*); HasProperty (*dessert, sweet*); MotivatedByGoal (*bake cake, eat*); SymbolOf (*Sacher Torte, Vienna*). These five relations altogether constitute approximately half of the assertions represented in ConceptNet 4.

The training datasets of each relation consist of approximately 500 assertions. Together we call them the 5-relation ConceptNet dataset in Table II above. Half of the assertions (SymbolOf is instantiated by 151 assertions only, and we used all) were randomly sampled from ConceptNet and the remaining assertions were constructed as bogus assertions by randomly picking an original assertion from the first half (e.g., *Sacher Torte SymbolOf Vienna*) and changing i) either one of its associated

Table IV. Meaningful and meaningless assertion decomposition of ConceptNet-based training datasets.

Relation	Meaningful	Total
AtLocation	148	452
IsA	150	477
HasProperty	158	516
MotivatedByGoal	151	450
SymbolOf	80	156

concepts with a random concept from ConceptNet (e.g., *Sacher Torte SymbolOf win election*), or ii) the original relation with another of the five relations we work with (e.g., *Sacher Torte IsA Vienna*).

For annotation of the training dataset, we recruited a total of 22 expert raters, all advanced students or researchers in artificial intelligence, semantics or related fields. The raters were given precise instructions on the purpose of the procedure and had to annotate assertions as *meaningful* or *meaningless*. For each rater, we computed the probability of agreement with the majority vote on a random assertion and, as a precaution to ensure high-quality data, we discarded the responses of five raters with a probability lower than 0.70. Only the 2,051 assertions which received at least two meaningful or two meaningless responses were considered for further analysis. The final label of an assertion was decided by the majority vote and the ties were broken in favor of meaninglessness. Table IV summarizes the annotation results for each relation. Note that some of the original assertions coming from ConceptNet were rated as meaningless (for example: *bread IsA put butter*; *praise IsA good job*; *read newspaper MotivatedByGoal study bridge*). These assertions should serve as high-quality negative instances given that they made their way into ConceptNet at one time as plausible assertions.

4.2. Candidate assertion mining

Unlike in the SAT and ConceptNet classification tasks reported above in Section 3 – where we are given a list of concept pairs in advance – to extract commonsense assertions from free text, we need a way to harvest *candidate* assertions, that can then be ranked by the algorithm.

The candidate assertions are mined from the syntactically parsed Wikipedia corpus made available by the WaCky project (see link above). The top 10,000 most frequent verbs, nouns and adjectives were considered as potential concept heads, and we extracted potential concept phrases with a simple grammar aimed at spotting (the content words of) noun, verb and adjective phrases (for example, the grammar accepts structures like `Adj Noun`, `Verb Adj Noun` and `Adv Adj`). In this phase, we were not interested in the semantic association between the concept pairs but simply tried to generate lots of pairs to feed to the trained BagPack models.

The pair extraction algorithm applied to Wikipedia produced 116,382 concept pairs. Then, we randomly sampled 5,000 pairs (containing 5,385 unique concept phrases) from this set, and generated 10,000 directed pairs by ordering them in both directions. Approximately 68% of the concepts in the sampled pairs were single words, 30% were 2-word phrases, 2% contained 3 or more words. Some example concept phrases that were mined are *wing*, *sport team*, *fairy tale*, *receive bachelor degree*, *father's death*, and *score goal national team*.

Finally, we associated the sample pairs with each of the five relations we study, obtaining a set of assertions that contain the same concept pairs, but linked by different relations. This step resulted in 10,000 candidate assertions for each relation.

4.3. Gold standard assertion set annotation

In order to provide a gold standard for further analysis, two expert raters annotated approximately 400 assertions for each relation (Wikipedia candidates in Table II above). Note that, unlike the annotated 5-relation ConceptNet dataset of section 4.1, the gold standard is based on actual new candidate assertions mined from Wikipedia. The sample was picked post hoc (i.e., after carrying

out the ranking procedure described in the next section), consisting of the candidate assertions that were ranked top by the BagPack models among the initial 10,000. The raters' overall Cohen's kappa was 0.37. The raters agreed on 183 meaningful assertions (8.8%) and 1508 meaningless assertions (72.6%). Any assertion that was annotated as meaningful by at least one rater was assumed to be meaningful for purposes of assessing performance, since the rate of meaningless assertions is so high that we reasoned it would be sufficient for an assertion to be at least potentially meaningful to be worth of further inspection. As a side note, the observed Cohen's kappa is very low compared to other tasks reported in the literature. However, in commonsense data annotation it is very hard to achieve higher agreement ratios. For example, in Schubert and Tong [2003], the reported mean kappa between pairs of raters on a 3-way decision task (involving categories "true", "false", and "undecidable") is 0.375. Experiments with other raters and more detailed guidelines lead us to kappas comparable to the one we are reporting, suggesting, together with Schubert and Tong's results, that we might be hitting an upper bound on human agreement about corpus-derived commonsense assertions.

4.4. Experimental setup

For each of the five datasets coming from the previous step, we trained a separate BagPack model. We extracted the co-occurrence vectors from the Web-derived English Wikipedia and ukWaC corpora as we did for the SAT analogy and ConceptNet classification task.

Since ConceptNet concepts are often expressed by multiple words (*Sacher Torte*, *eat too much*, ...), we employed a shallow search for the concept phrases. Basically, for a single phrase, we looked for the occurrence of the constituents with possible intermittent extra elements. We say a concept *occurs* in a sentence if all its constituents occur in the same order in the sentence with possible intermittent extra elements, and if the concept phrase spans no more than twice of its original length (e.g., if the concept is a 4-word phrase, it can span at most an 8-word range). Two concepts are said to be *co-occurring* if both concepts occur in the same sentence, they do not overlap (i.e., the last word of the first concept comes before the first word of the second concept), and the range that both concepts span together is not longer than 20 words (i.e., at most 18 elements can occur between the first word of the first concept and the last word of the last concept). For efficiency reasons, a maximum of 1,000 sentences were used to extract co-occurrence statistics for a given pair. The features in the co-occurrence matrix were weighted by PMI [Church and Hanks 1990]. The features were restricted to the most frequent 5,000 lemmas in ukWaC, resulting in a 15,000-dimensional vector for each pair.

We use the same settings for SVM implementation and training that we used in 3.2 (e.g., feature scaling, linear kernel, etc.)

4.5. Results

Similar to the ConceptNet classification task we used in Section 3.2, we report the area under the ROC curve (AUC). See examples of ROC curves for AtLocation in Fig. 3 below.

Table V reports the AUC obtained by the BagPack models trained on the ConceptNet-based training set (5-relation ConceptNet) and evaluated on the gold standard candidate assertions (Wikipedia candidates). For all relations, the performance of BagPack was significantly above the random baseline. However, AUC for MotivatedByGoal was barely above chance level and even the best AUC performance of 0.66 that was obtained on AtLocation was quite low, suggesting that BagPack alone cannot be used to extract reliable commonsense assertions from corpora despite its good comparative performance in classifying ConceptNet-extracted relations that we reported in Section 3.2 above.

Please note that compared to the AUC values given in Section 3.2, current values reported are considerably low. In the previous ConceptNet learning task we used a "closed" dataset that contains only original ConceptNet assertions whose labels are based on ConceptNet. Presumably, this is an easier dataset compared to the Wikipedia-based test set which contains assertions that are mined from Wikipedia.

Table V. BagPack AUC on five ConceptNet relations.

Relation	AUC
AtLocation	0.66
IsA	0.58
HasProperty	0.59
MotivatedByGoal	0.59
SymbolOf	0.58

5. THE CONCEPT GAME

The Concept Game[7] (CG) is a game with the purpose to collect common sense from laypeople. It is based on the idea that production of verbal information is a significant burden on the player and it is possible to design enjoyable games that do not require the players to produce assertions. Therefore, the game aims to achieve its purpose not by having the players produce commonsense assertions, but having them verify already collected candidate assertions. This approach allows us to design fast-paced games where the interaction between the user and the game is limited to an expression of assent. CG is presented in the context of a slot machine which produces random assertions. A meaningful assertion is a winning configuration. The trick is that the winning configurations do not dispense rewards automatically, but first they have to be recognized by the player to "claim their money". In this way, players tell us which assertions they found meaningful.

The game consists of independent play sessions each of which starts with an allocation of 40 seconds. First, the player sees three slots with images of rolling reels. They correspond to left concept, relation, and right concept of an assertion. Then, the contents of the slots are fixed one by one with some values picked from the database and as a result an assertion is displayed. At that point, the player has to press one of two buttons labeled as "Meaningless" or "Meaningful". If the player presses the Meaningful button this can result in two different outcomes: either the displayed assertion is indeed meaningful and (s)he is rewarded with two points and two bonus seconds (i.e., true positives are rewarded), or the assertion is in fact meaningless and the player loses three points and three seconds (i.e., false positives are penalized). However, pressing the meaningless button does not change the score or the remaining time (i.e., neither false negatives are penalized nor true negatives are rewarded). The feedback is conveyed to the player visually and acoustically (e.g., in case of a reward a green color flashes, in case of a penalty a red color flashes). The reels roll again, and the process repeats. This continues until the end of the allocated time, which can get longer or shorter depending on rewards and penalties. A typical screenshot of the game is given in Figure 1.

In the previous description, we pretended that the game already knows which labels are meaningful, and rewards or penalizes the user accordingly. This is not the case. In CG, we employ a validation policy similar to the honor-based, proof-of-payment method employed in many public transportation systems. In such a system, instead of checking every user, periodic controls are carried out to make sure the abuse of the system is effectively discouraged. In the current implementation, the probabilities of showing a known meaningless, a known meaningful, and a candidate assertion are 0.4, 0.3, and 0.3 respectively. In other words, 30% of the collected responses are for the new candidate assertions proposed by BagPack. For the candidate assertions, whatever the user responds is accepted as the correct answer and this is the actual knowledge we want to harvest. The meaningless assertions are used to verify the user is not abusing the game. The meaningful assertions are used to make sure the players score points and do not get frustrated. Note that increasing the precision of the candidate generation would help us to display more unknown assertions without a significant impact on the game experience (i.e. players would still be able to score points without the support of the meaningful assertions we display). To allow the players to warm up, we implicitly train them and do not show any unknown assertions until they complete three sessions with positive scores. Note that production of assertions that are known to be meaningless is relatively cheap. We can use the candidate assertions that are designated to be meaningless by the players (that's what

[7] http://apps.facebook.com/conceptgame/

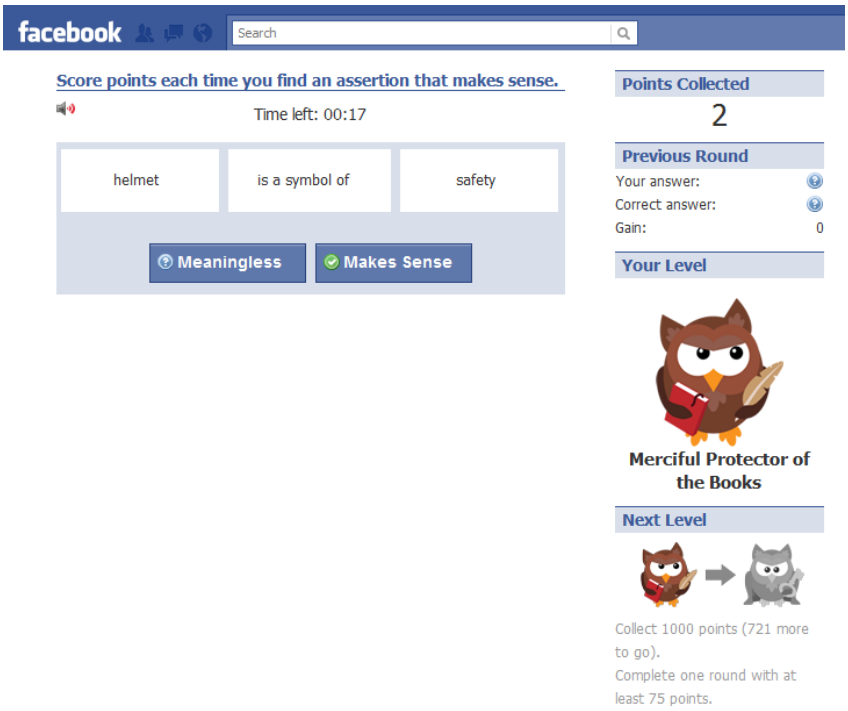

Fig. 1. Screenshot of a playing session in Concept Game

we do currently) or randomly combine concepts and relations to automatically generate assertions that are likely to be meaningless.

Technically, the game is almost equivalent to asking a group of raters to tick those assertions from a list which they think make sense. This is a dull task especially if there are few meaningful assertions compared to meaningless ones. In the context of a slot machine, however, the experience of seeing many meaningless assertions becomes part of the game, which creates an expectation in the player that (hopefully) resolves with a "winning" configuration. The relatively short session timing, combined with the need to be accurate because wrong claims are penalized, should keep the attention level of the players up, and consequently add to the fun. We made sure that players are aware of their achievements (they see total and session scores they have collected) and have an incentive to keep playing (we also display a top score list that shows the users who scored highest in a single session and we implemented a ladder system where the players are represented by cute avatars). Taking advantage of the integration with Facebook, we ask players' permission to post their activity in our game to their public walls and give them the opportunity to invite their friends in order to go up in the ladder.

As of September 2011, 1145 Facebook users had tried the application out of whom 278 passed the implicit training session and actually contributed to our dataset. After the first visit to the application page, 67% of the contributers returned and played the game at least on one other day while 36% returned at least on two other days. In total, we collected over 210,000 responses, approximately 65,000 of them for unknown, potentially meaningful assertions.

6. EXPERIMENTS WITH THE MINING+GAMING PIPELINE

In the following two experiments, we implement the mining+gaming pipeline and evaluate the quality of the assertions annotated by the players. In the first experiment, we kick-start the system by

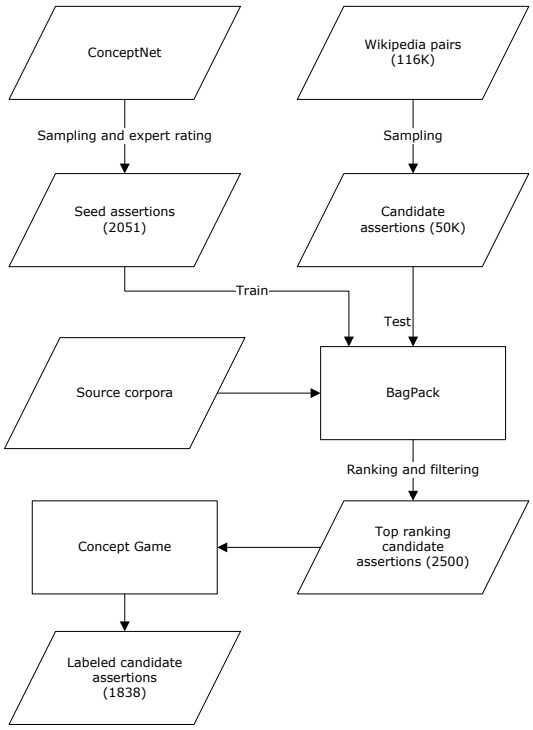

Fig. 2. Data flow for the kick-starting experiment. Concept pairs are extracted from Wikipedia as explained in Section 4.1. BagPack uses Wikipedia and ukWaC corpora to extract vector representations of assertions. Numbers in parentheses are the number of assertions in the datasets.

using the 5-relation ConceptNet dataset as the training data and compare the performance of BagPack alone and in combination with Concept Game in the common sense harvesting task. As a result, a new annotated dataset of commonsense knowledge is produced. In the second experiment, we bootstrap the combined system and use the output of the first experiment as the training data for a second round of BagPack training. Then, we compare the performance of the original BagPack models and the bootstrapped BagPack models on a new test dataset (i.e. Evaluation dataset in Table II).

6.1. Kick-starting

6.1.1. Experimental setup. The experimental procedure is summarized in Figure 2. The seed assertions based on ConceptNet and manually annotated as described in Section 4.1 serve as the BagPack training data (5-relation ConceptNet in Table II), and the set of assertions mined from Wikipedia serve as candidates to be ranked by the Concept Game. As already explained in Section 4.5, a separate BagPack model was trained for each relation and the candidates were ranked according to the confidence scores of these models. For each relation, we kept the top 400 ranking assertions as the set to be fed into the game (Wikipedia candidates in Table II). We will compare the performance of the labeling obtained from the game to the results we obtained by the "pure BagPack" approach in Section 4.5. Note that the gold standard for the Wikipedia candidates is obtained by the two expert raters annotation described in Section 4.3.

Once we ranked and filtered the candidate assertions by BagPack, we were ready to recruit players for their annotation via Concept Game. For this purpose, 18 people were contacted by email and were invited to play the game, mostly college students and staff that the authors personally knew. Unlike the raters used in the previous steps, players were *not* experts. The game was open to

Table VI. Summary of the Wikipedia candidate dataset. Labels are decided by majority votes based on CG players' responses with ties broken in favor of being meaningless.

Relation	Meaningful	Total
AtLocation	71	385
IsA	63	328
HasProperty	95	362
MotivatedByGoal	65	408
SymbolOf	73	355

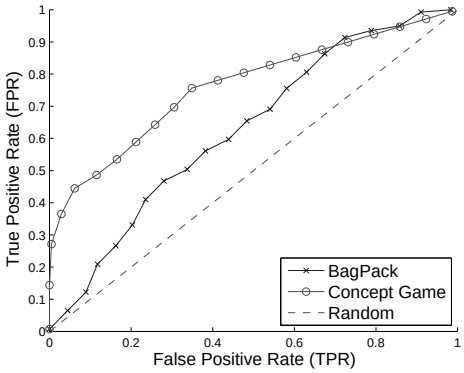

Fig. 3. ROC curves for AtLocation.

Table VII. Area under the ROC curve (AUC) on Wikipedia candidates.

Relation	CG	BagPack
AtLocation	0.76	0.66
IsA	0.79	0.58
HasProperty	0.69	0.59
MotivatedByGoal	0.72	0.59
SymbolOf	0.71	0.58

this "semi-public" for approximately 10 days. We used the negative training assertions for control purposes to penalize players' wrong decisions.

6.1.2. Results. In total, 25 players (7 presumably invited by the ones we contacted) responded and provided a total of 5,154 responses for the candidate assertions. The ratio of players who scored an assertion as meaningful was the *CG score* of the assertion. In addition to CG scores, we already had the BagPack confidence scores of the assertions.

In our analysis, we considered the 1,838 assertions (Wikipedia candidates) which received at least two meaningful or two meaningless responses, distributed across relations as shown in Table VI. The assertions that were labeled as meaningful consisted of 547 unique concepts and 86% of them were not attested in the 5-relation ConceptNet dataset (23% were not attested in the entire ConceptNet knowledge base). See Table II, for a summary of the datasets.

Using the expert raters' judgments as the gold standard for the candidate assertions, we computed relation-specific ROC curves for the CG. The areas under the ROC curves are given in Table VII – we repeat the BagPack's AUC values from the experiment in Section 4.5 for easier comparison. As an illustration, the ROC curves obtained for AtLocation are given in Fig. 3.

We observe that when the top BagPack candidate assertions are ranked by using the answers of the players, the performance considerably increases for all relations. This proves two points: First,

Table VIII. Meaningful and meaningless assertion decomposition of evaluation dataset.

Relation	Meaningful	Total
AtLocation	120	985
IsA	39	978
HasProperty	67	983
MotivatedByGoal	51	980
SymbolOf	41	973

BagPack alone is not sufficient to evaluate candidate assertions mined from Wikipedia reliably, and second, ConceptGame is able to improve performance. As an ad hoc evaluation, we computed the Cohen's kappa between the gold standard of the raters and the output of ConceptGame. The kappa value was 0.39 (comparable to the kappa value of 0.37 between the two raters themselves). Coupled with the high AUC values reported for the CG scores, we conclude that the annotation of the game players is comparable to manual annotation by experts.

6.2. Bootstrapping

In the previous experiment, we saw that combining BagPack and Concept Game can lead to a high-quality dataset of commonsense knowledge. So far, our criterion for quality has been the judgments of two expert raters on Wikipedia candidates dataset (see Table II. Now, we want to show that the output of the combined architecture can actually improve the performance in a more realistic task. Bootstrapping the entire system with its own output provides us with such a task. By bootstrapping, we mean training a new BagPack model only on the assertions mined from Wikipedia – with their labels decided by the Concept Game players. This experiment can be thought of as a continuation of the previous experiment where we use the labeled candidate assertions (the output of kick-starting) as the training seed of a new round of BagPack training, like adding a "train arrow" from the box at the bottom of Figure 2 to the BagPack box. We compare the performance of the bootstrapped BagPack and the original (i.e., ConceptNet-based) BagPack on a new evaluation set. That will allow us to see if the output of Concept Game is of sufficient quality to allow such a bootstrapping.

6.2.1. Experimental setup. In this experiment, we employ two different BagPack training seed assertions sets: The 5-relation ConceptNet dataset and the candidate assertions annotated by Concept Game in the previous experiment (i.e., the Bootstrap dataset summarized in Table II. For obvious reasons, the latter is called the *bootstrap* dataset from now on. In addition, we combined the two into a third dataset, the *combined* dataset. Descriptive statistics for the ConceptNet-based and bootstrap training sets were already given in tables IV and VI, respectively – the former as a result of expert annotation and the latter as a result of game playing.

To construct a final evaluation dataset (which will be called the *evaluation* dataset from now on), we randomly sampled approximately 1000 assertions for each relation by using the pairs mined from Wikipedia. We did not pick a set of candidates ranked by BagPack, as we did in the previous experiments, but used a random sample of assertions because we wanted to compare the performance of different BagPack models on the same evaluation set – ranking and filtering the assertions by any of the BagPack models would create a bias. We made sure, moreover, that, for each relation, the three corresponding BagPack seed datasets are disjoint with the evaluation set (i.e. they do not have any common assertions). The evaluation assertions were rated by the players of CG during a two-month period between April and May 2010 and they received at least two responses from different players. We already gave some details on this game playing session at the end of Section 5. For the annotation of the evaluation dataset, majority vote was used with ties broken in favor of being meaningless. The details of the evaluation dataset are given in Table VIII.

6.2.2. Results. In Figure 4, we report the area under the curve (AUC) values of the three BagPack models for each relation. The error bars represent the confidence intervals at the 95% significance level obtained by 5000 resamples with replacement.

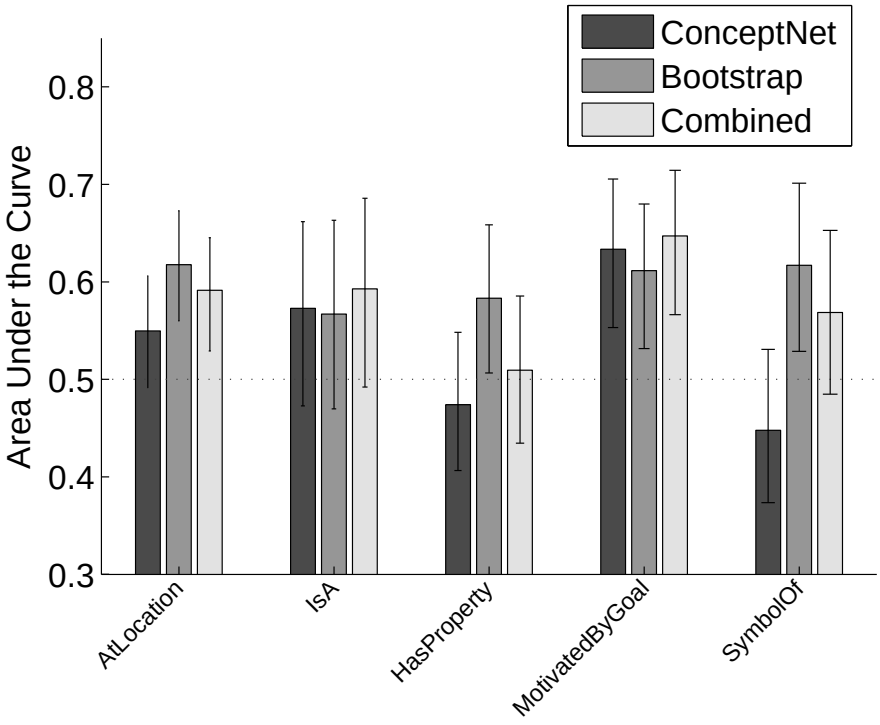

Fig. 4. AUC values for BagPack models trained on different datasets, for all relations. The dotted line at $y = 0.5$ is the chance-level performance. The error bars span the 95% confidence intervals. The gold standard is based on Concept Game players.

For AtLocation, HasProperty, and SymbolOf relations, bootstrapping obtains even better results compared to using original ConceptNet-based assertions. For IsA and MotivatedByGoal, bootstrapping is almost as good as using the original training dataset. The combination of two datasets brings additional improvements in IsA and MotivatedByGoal but the differences are not significant.

The main result of this experiment is that the data constructed with our method and no manual intervention, can be used as training data for a supervised algorithm that are as good, if not better, than expensive data obtained via expert annotation. Note that in this experiment, the gold standard for the evaluation dataset is provided by Concept Game, not by the experts, as the first experiment already showed that the output of Concept Game is of sufficient quality (see Bootstrap and Evaluation rows in Table II. The fact that both the bootstrap and evaluation datasets are annotated by the same process (i.e. Concept Game) may create an advantage for bootstrapping. Since the actual goal in this task is to find candidate assertions which are likely to be found meaningful by the players, we see this effect of shared annotation mechanism as an opportunity to be seized, not a nuisance variable to be controlled. Nevertheless, to see the extent of the effect, one of the authors manually annotated the entire evaluation dataset and we replicated the experiment with his ratings as the gold standard. The performance of the bootstrapped BagPack is not significantly worse than the original BagPack – its AUC values for the AtLocation, IsA, and HasProperty relations are higher than the original BagPack – though the differences are much less pronounced.

Another possible confounding factor we considered is the amount of overlap between the datasets in terms of concepts. The bootstrap dataset and the evaluation dataset come from the same population of assertions that are mined from Wikipedia. Therefore, even though they are disjoint in terms of assertions, they have a significant number of common concepts and that may be one of the rea-

sons why we obtain better results by bootstrapping. On the average, 21% of the unique concepts in the bootstrap dataset also occur in the evaluation dataset. In comparison, approximately 16% of the unique concepts in the ConceptNet-based dataset occur in the evaluation dataset. In order to control for this shared-concepts effect, we marked all concepts which are part of positive instances in the evaluation dataset and removed all assertions which contain these concepts from both training datasets. As a result, the total number of assertions in the bootstrap dataset reduced from 1838 to 1646 and the number of assertions in the ConceptNet-based dataset reduced from 2051 to 1892. Predictably, almost all computed AUC values are lower compared to the previous experiments with the untouched datasets. However, even though the decrease in AUC for the bootstrap dataset is visibly higher, bootstrapping results are almost as good as the original BagPack results. The fact that the gain of bootstrapping diminishes when we remove the common concepts from the training and evaluation sets is not discouraging because in real-world settings the bootstrap datasets will come from the same population of the future evaluation sets (as it does in our case), therefore a certain amount of overlap is what is to be expected.

Due to space limitations we do not report the figures related to these two control experiments.

Summarizing the second experiment, the bootstrap dataset – which was mined from Wikipedia, filtered by BagPack and annotated by the 25 Concept Game players – is at least as successful as the 5-relation ConceptNet dataset – which depends on ConceptNet and was annotated by 22 experts – in seeding BagPack to extract commonsense knowledge from corpora. We conclude that, in our task of training BagPack to extract commonsense knowledge, the output of Concept Game can be used instead of a dataset that is created by expert raters. This in turn bids well for application of BagPack and Concept Game in other AI tasks requiring quality commonsense data.

7. CONCLUSION

We showed how two different approaches to knowledge extraction can work in tandem and produce a new dataset of high precision commonsense knowledge. The combined mining+gaming architecture was able to produce significantly better commonsense facts than our quality text miner alone. Furthermore, we report that bootstrapping the system with its own output improves the performance. In our case, bootstrapping amounts to reducing the need for expensive expert-annotated data to a first seeding of the system.

In addition, we do not have to rely on the human contributors for the creation of new knowledge but use the corpus (via the text miner) as the source. Presumably, that allows us to collect assertions about concepts that humans do not tend to state explicitly. This advantage was reflected in the fact that 23% of the concepts that we mined were not attested in the entire ConceptNet knowledge base.

Concept Game is a fully functional and public Facebook application. In the short term, we are looking for ways to make the game more attractive to a wider non-specialized audience. We would like to convert the lemma sequences produced by BagPack into natural sounding sentences. We have recently started to offer small gifts to top players as an incentive to start and keep playing.

A recent (and raw) snapshot of the data we collected is downloadable from the web[8]. Once we gain a reasonably wide player-base and construct a larger dataset of commonsense assertions we plan to share the dataset in a more structured form.

In future analyses, we would also like to look for cultural differences in assertions that receive contrasting ratings from players from different continents. Using Facebook as our platform allows us to access demographics of players for statistical analysis.

While these and many other avenues of development and analysis should be pursued, we believe that our current results make a strong case for the feasibility of an approach that mixes text mining and social intelligence to harvest commonsense knowledge on a large scale.

[8] http://github.com/amacinho/Concept-Game-Datasets

REFERENCES

AIROLA, A., PAHIKKALA, T., WAEGEMAN, W., DE BAETS, B., AND SALAKOSKI, T. 2011. An experimental comparison of cross-validation techniques for estimating the area under the ROC curve. *Computational Statistics & Data Analysis 55,* 4, 1828–1844.

BANKO, M., CAFARELLA, M., SODERLAND, S., BROADHEAD, M., AND ETZIONI, O. 2007. Open information extraction from the web. In *Proceedings of the International Joint Conference on Artificial Intelligence*. Hyderabad, India, 2670–2676.

BARONI, M. AND LENCI, A. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics 36,* 4, 673–721.

BICIÇI, E. AND YURET, D. 2006. Clustering word pairs to answer analogy questions. In *Proceedings of the Fifteenth Turkish Symposium on Artificial Intelligence and Neural Networks*. Muğla, Turkey, 277–284.

BUITELAAR, P. AND CIMIANO, P. 2008. *Bridging the Gap between Text and Knowledge*. IOS, Amsterdam.

CANU, S., GRANDVALET, Y., GUIGUE, V., AND RAKOTOMAMONJY, A. 2005. SVM and Kernel Methods Matlab Toolbox.

CARLSON, A., BETTERIDGE, J., KISIEL, B., SETTLES, B., HRUSCHKA JR, E., AND MITCHELL, T. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence*.

CHURCH, K. AND HANKS, P. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics 16,* 1, 22–29.

ESLICK, I. 2006. Ms thesis. Ph.D. thesis, MIT, Cambridge, MA.

FAWCETT, T. 2006. An introduction to roc analysis. *Pattern Recogn. Lett. 27,* 8, 861–874.

GORDON, J., DURME, B. V., AND SCHUBERT, L. 2010. Evaluation of commonsense knowledge with mechanical turk. In *In NAACL Workshop on Creating Speech and Language Data With Amazons Mechanical Turk*.

GORDON, J., VAN DURME, B., AND SCHUBERT, L. 2010. Learning from the web: Extracting general world knowledge from noisy text. In *Proceedings of the AAAI 2010 Workshop on Collaboratively-built Knowledge Sources and Artificial Intelligence*. ACM.

HAVASI, C., SPEER, R., PUSTEJOVSKY, J., AND LIEBERMAN, H. 2009. Digital intuition: Applying common sense using dimensionality reduction. *IEEE Intelligent Systems 24,* 4, 24–35.

HERDAĞDELEN, A. AND BARONI, M. 2009. BagPack: A general framework to represent semantic relations. In *Proceedings of the EACL GEMS Workshop*. Athens, Greece, 33–40.

LENAT, D. 1995. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM 11*, 33–38.

LIEBERMAN, H. 2008. Usable AI requires commonsense knowledge. In *Workshop on Usable Artificial Intelligence, ACM Conference on Computers and Human Interaction (CHI-08), Florence, Italy*.

LIEBERMAN, H., SMITH, D., AND TEETERS, A. 2007. Common consensus: a web-based game for collecting commonsense goals. In *Proceedings of IUI07*. Honolulu, HI.

LIU, H. AND SINGH, P. 2004. Commonsense reasoning in and over natural language. In *Knowledge-Based Intelligent Information and Engineering Systems*. Springer, 293–306.

MATUSZEK, C., WITBROCK, M., KAHLERT, R., CABRAL, J., SCHNEIDER, D., SHAH, P., AND LENAT, D. 2005. Searching for common sense: Populating cyc from the web. In *Proceedings of the National Conference on Artificial Intelligence*. Vol. 20. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 1430.

MCCARTHY, J. 1959. Programs with common sense. In *Proceedings of Teddington Conference on the Mechanization of Thought Processes*. London, England, 75–91.

MINSKY, M. 2000. Commonsense-based interfaces. *Communications of the ACM 43,* 8, 66–73.

QUESADA, J., MANGALATH, P., AND KINTSCH, W. 2004. Analogy-making as predication using relational information and LSA vectors. In *Proceedings of CogSci*. Chicago, IL, USA, 1623.

QUINN, A. J. AND BEDERSON, B. B. 2009. A taxonomy of distributed human computation. Tech. Rep. HCIL-2009-23, University of Maryland, College Park.

RAFELSBERGER, W. AND SCHARL, A. 2009. Games with a purpose for social networking platforms. In *HT '09: Proceedings of the 20th ACM conference on Hypertext and hypermedia*. ACM, New York, NY, USA, 193–198.

RAMACHANDRAN, D., REAGAN, P., AND GOOLSBEY, K. 2005. First-orderized researchcyc: Expressivity and efficiency in a common-sense ontology. In *AAAI Workshop on Contexts and Ontologies: Theory, Practice and Applications*.

SCHUBERT, L. AND TONG, M. 2003. Extracting and evaluating general world knowledge from the Brown corpus. In *Proceedings of the HLT-NAACL 2003 workshop on Text meaning-Volume 9*. Association for Computational Linguistics Morristown, NJ, USA, 7–13.

SMITH, B. 1995. Formal ontology, common sense and cognitive science. *International Journal of Human Computer Studies 43,* 5, 641–668.

SPEER, R. 2007. Open Mind Commons: An inquisitive approach to learning common sense. In *Proceedings of the Workshop on Common Sense and Intelligent User Interfaces*. Honolulu, HI.

SPEER, R., HAVASI, C., AND SURANA, H. 2010. Using verbosity: Common sense data from games with a purpose. In *Proceedings of the Twenty-Third International Florida Artificial Intelligence Research Society Conference (FLAIRS 2010)*. 104–109.

STROHMAIER, M. AND KRÖLL, M. 2009. Studying databases of intentions: do search query logs capture knowledge about common human goals? In *Proceedings of the fifth international conference on Knowledge capture*. ACM, 89–96.

TURNEY, P. 2006a. Expressing implicit semantic relations without supervision. In *Proceedings of COLING-ACL*. Sydney, Australia, 313–320.

TURNEY, P. 2006b. Similarity of semantic relations. *Computational Linguistics 32,* 3, 379–416.

TURNEY, P. 2008. A uniform approach to analogies, synonyms, antonyms and associations. In *Proceedings of COLING*. Manchester, UK, 905–912.

TURNEY, P. AND LITTMAN, M. 2005. Corpus-based learning of analogies and semantic relations. *Machine Learning 60,* 1-3, 251–278.

TURNEY, P., LITTMAN, M., BIGHAM, J., AND SHNAYDER, V. 2003. Combining independent modules in lexical multiple-choice problems. *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*, 101–110.

VANDERWENDE, L. 2005. Volunteers created the web. In *2005 AAAI Spring Symposium on Knowledge Collection from Volunteer Cotnributors (KCVC'05)*.

VON AHN, L. 2006. Games with a purpose. *Computer 29,* 6, 92–94.

VON AHN, L., KEDIA, M., AND BLUM, M. 2006. Verbosity: A game for collecting common-sense knowledge. In *Proceedings of CHI*. Montreal, Canada, 75–78.

WRIGHT, M., CHODZKO, J., AND LUK, D. 1996. *HAL's Legacy: 2001's Computer as Dream and Reality*. MIT Press, Chapter 9, 193–209.

YU, C.-H. AND CHEN, H.-H. 2010. Commonsense knowledge mining from the web. In *Proceedings of AAAI 2010*.