Activity Inference through Commonsense

Kun Tu and **Megan Olsen** and **Hava Siegelmann**
Department of Computer Science
University of Massachusetts Amherst
140 Governors Drive, Amherst, MA 01003

Abstract

We introduce CIM, a Commonsense Inference Memory system utilizing both Extended Semantic Networks and Bayesian Networks that builds upon the commonsense knowledgebase ConceptNet. CIM introduces a new technique for self-assembling Bayesian Networks that allows only relevant parts of the commonsense database to affect the inference. The Bayesian Network includes both the activities occurring within the input sentences and the related activities appearing in the commonsense database. The Bayesian Network is used to interpret and infer the meaning of the set of input sentences. With our self-assembled networks only relevant inference is performed, speeding up performance of reasoning with commonsense knowledge. We demonstrate that our system can disambiguate the needs of the user even if they do not state them directly, and do not use keywords. This ability would not be possible without either the use of commonsense or significant training. Eventually this approach may be applied to increase the effectiveness of other natural language understanding techniques as well.

Introduction

Natural Language Processing studies techniques to understand the semantics of written language in general documents (Ali and Shapiro 1993; Jurafsky, Martin, and Kehler 2000). Methods for understanding semantics are usually combined with syntax to increase understanding. The use of commonsense reasoning in natural language processing has so far been limited, however. One example of using commonsense reasoning is for inference in story understanding. Previous approaches to story understanding applied language processing operations to build effective reasoning models. Commonsense methods have been designed for domain specific reasoning such as reasoning on the cause of a car accident(Kayser and Nouioua 2009), or generating summaries according to data in medical care (Portet et al. 2009). Others aimed for a more general approach for using commonsense in NLP. For instance, the search engine Goose uses commonsense to understand a user's search query without relying on keywords (Liu, Lieberman, and Selker 2006). Another general method establishes co-reference mappings of data in a memory system to reduce the number of ambiguous sentence interpretations and determine whether different references describe the same event (Livingston and Riesbeck 2009). This approach may often be slow as it focuses solely on matching entities and stories through memory. We instead focus our approach on understanding language through activities. Our approach is based on the fact that the semantic roles of verbs have been used to characterize nouns, and have been shown to predict the brain activity associated with the meanings of nouns (Mitchell et al. 2008). By focusing on the action components of sentences, our commonsense inference memory (CIM) can improve inference with commonsense by associating related commonsense activities with those in the input sentences.

There are several commonsense knowledge bases that can be used for language reasoning, such as Cyc (Lenat, Prakash, and Shepherd 1985) from Cycorp and ConceptNet (Liu and Singh 2004) from the Open Mind Common Sense Project. We have chosen to use ConceptNet due to the ease of integrating ConceptNet with our CIM system. In ConceptNet, a word from the input sentences is given and then related concepts are found in the commonsense database and returned. Learner has improved on the techniques used in ConceptNet by creating a system that is similar to ConceptNet, but with the addition of analogies to suggest new topics (Chklovski 2003). Although they are able to suggest additional knowledge, their results state that just over 50% of their suggestions are incorrect or irrelevant. Similar to but more mathematically rigorous than Learner, AnalogySpace calculates new potential analogy based inferences between commonsense concepts of similar features (Speer, Havasi, and Lieberman 2008). We provide a more general way to create new inferences that allows the system to work toward understanding different sets of input over time through Bayesian inference as opposed to the more limited analogy inference.

Similar to these systems, we take into account the type of information we are analyzing, disambiguate word meanings, and find the sentence topics prior to

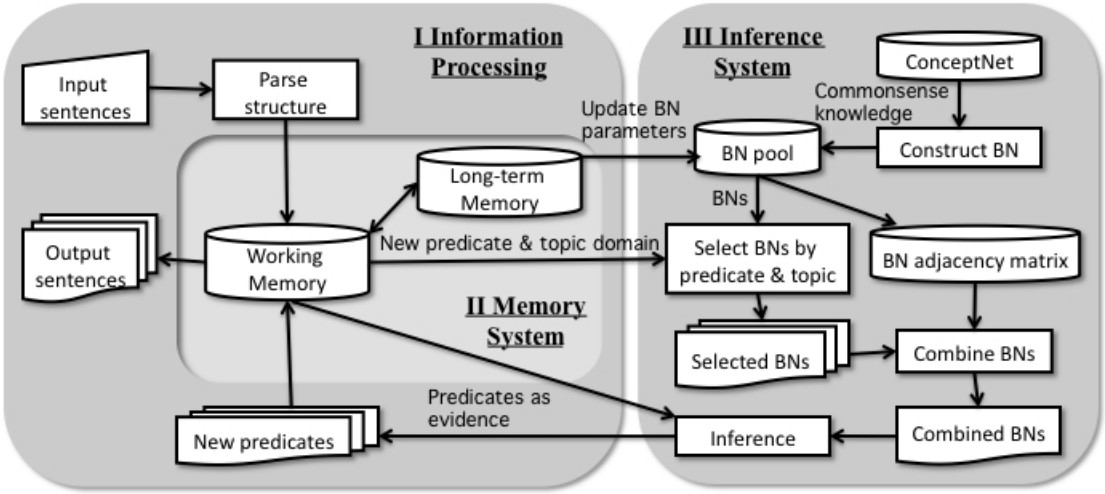

Figure 1: An overview of the Commonsense Inference Memory (CIM) system, consisting of three subsystems: Information Processing, Memory, and Inference. The information process system first analyses input sentences to determine the topic and disambiguate meanings (details in Fig.2). The input is stored in the Memory system, and then sent to the Inference system. The inference system combines the input with commonsense knowledge that has been stored in the form of small 2-node Bayesian networks. This combination creates larger networks, linking commonsense knowledge with domain knowledge from the input. These larger networks are then returned to memory where they can be used to continue inferring new knowledge as sentences are input, as well as to create new relevant information.

inference to ensure only accurate new knowledge is created. We introduce the Commonsense Inference Memory (CIM) system to understand language using commonsense knowledge. It parses input sentences with a modified version of the Stanford parser (de Marneffe, MacCartney, and Manning 2006). The information is refined via Wordnet (Fellbaum 1998), VerbNet (Kipper et al. 2006), and the ConceptNet database, and then stored for inference. Assertions in ConceptNet are used to create two-node Bayesian networks (BNs) that display causal relations of two activities. Related BNs are adaptively selected by matching the topic keywords given by these new input sentences, and are then combined with common nodes. This automated selection and assembly of BNs enables inference processes to avoid inaccurate conclusions and to potentially combine commonsense knowledge better than in previous models (Chklovski 2003; Speer, Havasi, and Lieberman 2008). We demonstrate the use of our system on understanding user needs through natural language statements, without the use of keyword commands. We show that CIM can use commonsense knowledge to understand user needs by examining actions in a novel way that may not be possible without commonsense.

An Overview of the CIM System

Our system is a memory system using Extended Semantic Networks (ESN) to infer new information via a reasoning model of self-assembled Bayesian networks. Fig. 1 shows an overview of our model.

The memory system receives input information in the form of natural language sentences. These sentences are first parsed into subjects and predicates using the Stanford Parser. Word meanings are disambiguated by referring to constraints in VerbNet or by matching topic keywords generated from WordNet. Fig. 2 shows the procedure of analyzing sentences. The parsed information is stored in a temporally built ESN that acts as working memory. Relevant information in our database (long-term memory) is also added to the working memory to assist in comprehending the new information.

The next step is to construct a reasoning model using commonsense knowledge. A Bayesian network (BN) is suitable to infer new information by integrating the existing facts and commonsense knowledge. In Conceptnet, knowledge is represented as a relation between two concepts or events. These events can be viewed as nodes in a BN and thus we can construct well-structured BNs with these relations. Information from the input can also be converted into events in a commonsense-based BN and serve as evidence during the calculation of probabilities. A BN adjacency matrix is built to indicate if two BNs share the same node. When new data arrives from the working memory, BNs are selected by matching the predicates and topic domains. Then the BNs are combined according to the adjacency matrix. Finally, the combined BNs use evidence from the working memory and calculate the probability of the BN's predicates. Predicates with high probability are regarded as new information and propagated to the working memory. More details follow in the rest of the paper.

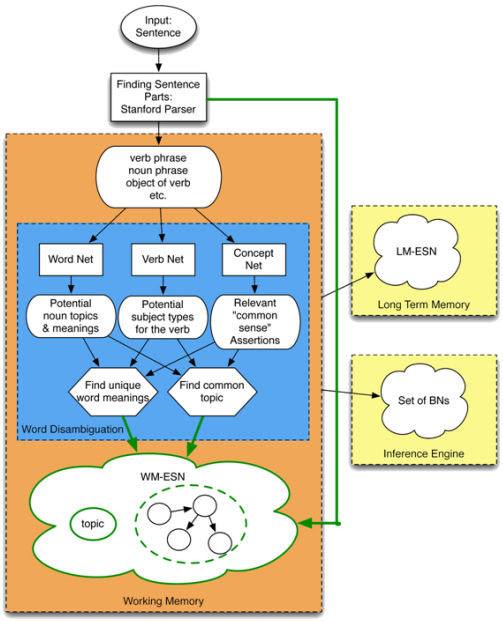

Figure 2: Information Processing System: Analyzing information from the input. Sentences are first parsed with the Stanford Parser and stored in a temporarily built ESN. Then, word meanings are disambiguated: topic keywords and word meanings from WordNet are selected with the constraints from VerbNet and ConceptNet to decide the possible topic domain of the sentences. This refined information is then propagated to the long-term memory or inference engine.

Information Processing

Parsing Sentences into Working Memory

The parsing module provides formatted information to integrate with commonsense data (Fig. 2). It parses input sentences and breaks them into several parts based on grammar. Predicates in these sentences describe the features or activities of the subjects. Some predicates can break into the form of "joiner + object," where the joiner contains a verb, indicating the relations between the subjects and the objects (Fig. 3(a)). From a graphical point of view, a vertex can symbolically represent a subject or an object while an edge can represent their relation. These extended semantic networks (ESNs) are used to represent the information derived from sentence, as shown in Fig. 3. Notice that there are differences between the ESN and classical semantic networks: 1) vertices in the ESN only represent subjects and objects, and edges represent the relations between vertices; 2) An edge has a probability if the predicate that it belongs to comes from inference, however, if an edge comes directly from an input sentence, it is labeled in the ESN as evidence for inference; 3) There are multiple directed edges from one vertex to another; 4) a loop edge is allowed.

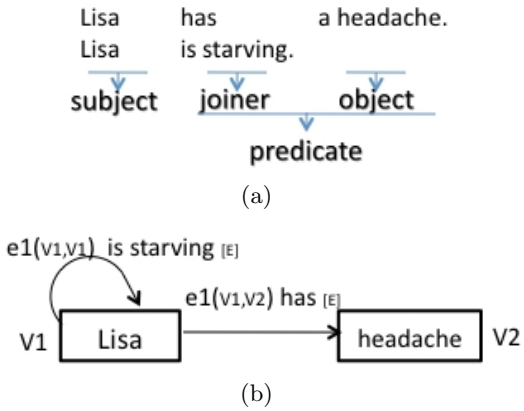

Figure 3: Parsing sentences into the ESN. (a) Two sentences are divided into subjects and predicates and the predicates are broken into joiner and object. (b) Subjects and objects in the sentences are stored as vertices in the ESN; joiners are stored as edges connecting two vertices. The joiners "is hungry" and "has" are marked as evidence since they come from the input sentences.

Word Disambiguation

After the sentences are parsed, the meanings of words must be disambiguated so that the input is not misunderstood in current or future reasoning. After the input sentences are parsed into subjects, joiners and objects, WordNet and VerbNet are used to disambiguate word meanings and ensure the accuracy of information by topic matching. WordNet provides meanings of words and relations between synsets such as "is a kind of," "is similar to," and "topic domain." For example, WordNet provides the first step toward distinguishing the word "bat" in the sentence "a bat eats insects," from several potential meanings: 1)"club used to hit balls in a ballgame," 2) "nocturnal mouse like mammal," and 3)"small racket with long handle to play squash." Meanwhile, VerbNet has semantic restrictions for verbs to constrain their thematic roles. The transitive verb "eat" requires a living object such as an animal or human to perform the activity.

Another technique for word disambiguation is to match topic keywords of sentences. For example, the sentence "he goes to the bank" has ambiguous meanings: "he goes to a sloping land beside the water" or "he goes to a depository financing institution." The meaning of the sentence cannot be disambiguated only with WordNet and VerbNet. If there is another sentence "he jumps into the water," the working memory then gets two potential topic domains from word meanings in WordNet: ["sloping land, water", "water"] and ["depository, financing institution", "water"]. Only the first topic domain shares the common word "water," thus the meaning of "bank" is chosen to be a "sloping land" because the meaning can be categorized to the same topic with the second sentence.

With these disambiguation techniques, words in the sentences can be assigned specific meanings. These meanings are later used to select the related knowledge by matching their topic words and feature words, after being stored in memory.

Sharing Information through Memory

Two parts constitute the memory model that facilitates information sharing between the Information Processing and Commonsense Reasoning, both represented by an ESN: working memory (WM), and long-term memory (LTM). The WM is a temporarily created ESN that only stores information related to current reasoning. This information includes the most recent input sentences, inferred data from the reasoning model, and related information from the LTM.

At the end of the inference process relevant data is stored in the LTM for later use. The LTM stores both previously input and inferred data. When subjects of new sentences arrive in the WM, input data of the same subjects in the LTM are searched and copied to WM as additional information for inference. This data is also used as training data to update parameters such as the conditional distribution in the Bayesian Networks in our reasoning model.

One problem in a reasoning process is that there can exist different pieces of input evidence causing a contradiction of conclusion. For example, if someone "works hard," then they would "feel tired." On the other hand, if they "rest," then they would no longer "feel tired." This change could cause a contradiction if the person is observed to "work hard" and "rest" in the evidence because they could still "feel tired" without enough rest. One solution to this type of problem is to use probabilities to decide the dominant factor ("work hard" or "rest") to a conclusion. Thus, probabilities allow the system to easily accept options, and we apply a Bayesian network for reasoning.

Inference with Commonsense

CIM understands language by combining the relevant information in the ESN with the commonsense data stored in small Bayesian networks before performing inference. By creating small two-node Bayesian networks of commonsense knowledge initially, and then combining them with input from the ESN when necessary, we decrease the time it takes to reason using commonsense.

Bayesian Network Constraints

To ensure the communication between the inference module and language parsing module, our system places several constraints on the Bayesian Network so that information in the ESN can be easily integrated for inference. The first constraint is that each BN node only represents an activity or a property of a subject of a sentence. Thus, BNs in our system can receive evidence that is converted from the ESN for probability calculation, as the ESN stores information about subjects.

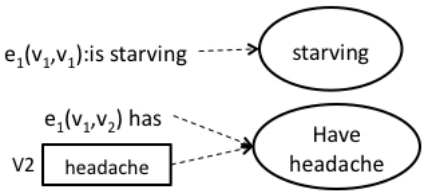

Figure 4: Conversion of information from ESN to BN evidence. Starting from the ESN information in Fig. 3(b), the edges and nodes are converted to BN nodes.

In the conversion, an edge representing an unary relation in an ESN changes directly into a variable, denoted as a BN node. Its probability distribution is initialized, where values of the variable are "true" and "false" (Fig. 4, edge $e_1(v_1, v_1)$). A directed edge representing a binary relation in an ESN is combined with its pointed node into a BN node (the edge $e_1(v_1, v_2)$ and the node v_2 in Fig. 4).

The second constraint is that the activity and property of node in a single BN should have the same agent. This is different from a traditional BN, where each node can represent events for different agents. This constraint ensures the system will infer new information for the correct agent.

Bayesian Construction from Commonsense

To infer new information from a parsed sentence, Bayesian networks are constructed from commonsense knowledge to update activities of entities based on new information (Fig. 1). The initial preprocessing stage creates two-node Bayesian Networks from ConceptNet. The knowledge in ConceptNet is expressed as five-tuple assertions ("relation type", "A", "B", "f", "I"), where "relation type" indicates the relation of concept "A" and "B," f is the number of times a fact is uttered in their training corpus, and i counts how many times an assertion was inferred during the 'relaxation' phase. For example, (CapableOf "animal" "grow" "f=2; i=2;") has a relation type "CapableOf," which indicates that if something is an "animal," it is capable of performing the activity "grow." We chose nine of the twenty relation types to create our initial two-node BNs(as seen in Table 1). These nine relation types are the most relevant, as they are all related to activities. For example, "CapableOf" will aid in classifying a person based on what they do. However, "MadeOf" or "PartOf" will not aid our classification as both will only give us information about the physical properties of an item, and does not relate to its use in activities. With this set, all nodes in the self-assembled Bayesian networks continue to relate to actions only.

Additionally, we need to initialize the parameters of the newly constructed BNs since the joint conditional distributions of the BNs are not provided in ConceptNet. We initialize the probability distribution values according to the relation type of the corresponding as-

sertion (Table 1). The probabilities of nodes without a parent are set to 0.5.

However, the parameters of the BNs can be later adjusted according to training data from the LTM. If there are 10 people in the LTM with the activity "have lots of cholesterol" and 7 out of them also have the activity "have heart attack," then the probability $P(have\_cholesterol \mid have\_heart\_attack)$ is adjusted to 7/10. Deciding the probability function is a task belonging to the structure learning of BNs. Much work has been done on creating probability functions with training data (Niculescu-Mizil and Caruana 2007; Tsamardinos, Brown, and Aliferis 2006), which we utilize in our system.

Bayesian Network Combination

When the newly parsed information is acquired from the Working Memory (WM), BNs are selected and joined together into a larger Bayesian net for activity understanding (Fig.1). The selection process includes two main steps: 1) selecting the BNs that contain exactly the same information as the input and 2) selecting BNs containing relevant information.

The first main step of selection is based on matching BN nodes, predicates in the sentences, and the topics in the working memory. After searching for BNs with nodes representing the same predicates as the sentences, select BNs containing the predicates that share the same topic domain as from the working memory. For example, if there is a sentence "he goes to the bank" in the working memory and the topic keywords are "depository, financing institution," BNs with the node "go to the bank" will be selected. Then BNs with the same topic keyword remain but BNs with other topic keywords such as "sloping land, water" will be removed.

At this point, multiple two-node BNs will have identical nodes representing the same predicate. We will combine these BNs so that they share the same node, and the length of the longest path in the new BN is two. Thus, the new BN can only infer information with a direct causal relation from the input sentence. To infer new information, the second step of the selection obtains relevant BNs that do not have the same predicates as the input but are related.

An adjacency matrix M is built for selecting such

relation type	$P(A\|B)$	$P(A\|\neg B)$
(PrerequisiteEventOf "A","B")	0.9	0.2
(FirstSubeventOf "A","B")	0.7	0.3
(EffectOf "A","B")	0.7	0.2
(CapableOf "A","B")	0.7	0.1
(SubeventOf "A","B")	0.9	0.1
(MotivationOf "A","B")	0.6	0.1
(DesirousEffectOf "A","B")	0.6	0.4
(IsA "A","B")	1.0	0.6

Table 1: Default probabilities for integrating information to a BN from ConceptNet.

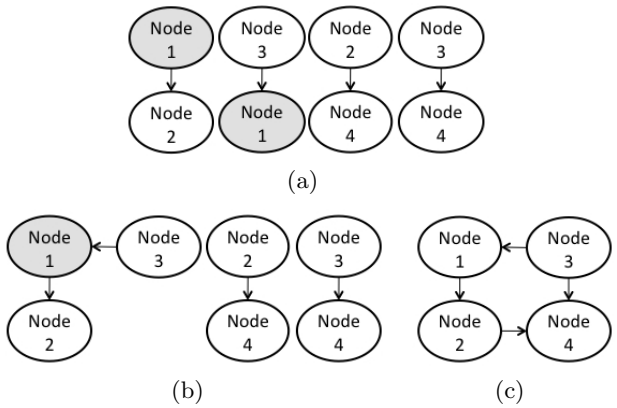

Figure 5: Combination of two-node BNs. (a) Four selected BNs, where node 1 represents the information from the ESN. The two leftmost BNs are selected in the first step by matching the information, that they share node 1. (b) The two BNs with node 1 have been combined. Now the other matching nodes are selected by searching a d-connecting path in the adjacency matrix. (c) After iterating through the process of combining nodes, we have a combined BN of four nodes.

BNs. If BN i and BN j share a common predicate, then $M_{ij} = M_{ji} = 1$, otherwise $M_{i,j} = 0$. By multiplying the adjacency matrix with itself, a path connecting the BNs can be found. By searching d-connecting paths containing the nodes that represent the input information, all the relevant BNs can be selected. This saves us from learning and inferring with unrelated data, and hence reduces the complexity and increases the accuracy in over typical BN strategies.

After the selection, BNs are joined together according to the shared node s. The joint distribution of each node in the BN needs to be adjusted according to the newly combined BN structure. Suppose the shared node s have the set of parents a_i in BN i and a_j in BN j, where $\forall x, x \in a_i \longleftrightarrow x \notin a_j$. From the training data in the longterm memory, we can calculate the conditional distributions $P(s \mid a_i)$ and $P(s \mid a_j)$. The new joint conditional distribution $P(s \mid a_i, a_j)$ can be modified in two ways: 1) assign the value of the distribution manually or 2) calculate the distribution according to the relations of the variables in a_i and a_j . As the first method is trivial and impractical, we must define the new distribution using method 2. We accomplish this calculation as seen in Eq. 1 and Eq. 2.

$$P(s \mid a_i, a_j) = \sum_m P(s \mid a_i, a_{j,m}) \frac{P(a_{j,m}, a_i)}{P(a_i)} \quad (1)$$

$$P(s \mid a_i, a_{j,m}) = \sum_k P(s \mid a_{i,k}, a_{j,m}) P(a_{i,k} \mid a_{j,m}) \quad (2)$$

where $a_{i,k}$ refers to the k^{th} parent of s in BN i. The joint distribution $P(a_{j,m}, a_i)$ can be obtained by searching a d-connecting path through the BNs and then the conditionals $P(a_i \mid a_{j,m})$ and $P(a_{j,m} \mid a_i)$ can be calculated.

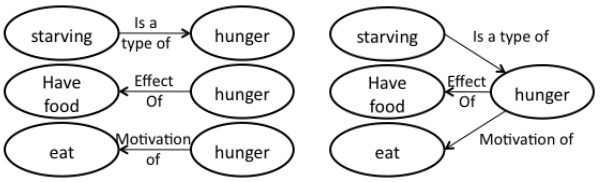

Figure 6: BN Combination based on nodes from Fig. 4. (a) Some of the potential nodes from conceptnet. (b) Since we know we want to know more about starving, we find the connection between starving and hunger. Then we find the nodes connected to hunger, leading us to our answer.

If there is no such d-connecting path, the variables are considered to be independent.

The assembled BNs use the predicates in the new input sentences as evidence to calculate the probabilities of other nodes. The predicates, represented by nodes that have high probabilities, are then stored in the working memory as new data. This new data is updated to the long-term memory, and also used for creating output.

Demonstration

Often in automated language analysis it can be difficult to determine implications, as they usually rely on the listener having commonsense. In the case of a computing system interacting with elderly users this can become a bigger problem as they may not be able to memorize keywords to give the system, or may not feel comfortable interacting with computer systems. Thus, it is beneficial for the system to be able to interpret user needs through natural language.

Take for example the activity of needing to eat. A user may have a variety of ways to state that they are hungry: "I'm starving," "I want a snack," "Is it lunch time yet," or "Are there any leftovers" to name a few. None of these phrase incorporate a keyword such as "eat" or "food" and there are many more examples that also do not use such a keyword. However, as humans we understand the semantics of these words easily.

Our system can also understand these phrases in many cases. For example, in the case of Lisa being hungry seen in Figures 3 and 4, we can combine knowledge of ConceptNet with our input about Lisa to determine that she wants to eat (Fig. 6). If nodes with the necessary information are further away, additional iterations will still connect them to the input node. Multiple iterations are key to the success of the system, as it allows inferring information that would not be found through other techniques. However, to guarantee speed we limit the number of iterations performed, which does not significantly impact our ability to understand sentences. Thus, we can analyze input to determine the needs of the user by utilizing commonsense.

Conclusion

The CIM system can generate new sentences about a subject based on a combination of the input sentences, previous experience, and commonsense knowledge. Commonsense knowledge from ConceptNet is first obtained as two-node BNs, focused on the activities occurring within the sentences. These BNs can adaptively combine into a larger BN by finding a path between them via the adjacency matrix in an iterative manner. The process creates a single BN containing all (and only) relevant paths from the input sentences to the query predicate. This BN can be translated to an ESN for the creation of new sentences, giving new inferred information about our input.

This model uses the long-term memory ESN to store incoming sentences as previous experience that can later become the training data to determine the parameters of the BNs. This allows the probability distributions of the BNs to be dynamic in such a way that new probabilities are automatically obtained each time a new scenario causes the BN combination to occur. In addition, it allows the system to learn from its input. By creating a system combining ConceptNet with Extended Semantic Networks and multiple Bayesian networks that are capable of assembling, we are able to improve on current inference from input sets of sentences. We improve on the classical Bayesian Network by considering only smaller well-focused Bayesian Networks, which are thus faster for inference. This can significantly reduce the cost of Bayesian inference. In addition, we are able to provide improved inference over ConceptNet, as its result cannot be modified given the same input information. Therefore, CIM provides improved language understanding based on commonsense reasoning that can be applied in human-computer interfaces and other natural language processing problems.

We have shown a way to organize commonsense information so that it can be used to infer new information and new sentences based on actions from the input. Although there are some immediate applications to user interaction scenarios, we also plan to investigate the integration of our work with more standard NLP techniques. We suggest first inferring new activities using our system to create a set of new sentences that can be combined with the original input sentences to create a larger set of sentences. Then we can apply existing language understanding techniques to this enlarged set of input sentences, which could then potentially improve the results of those techniques such as topic models. Thus, our system has the potential to also increase the success of more generally used NLP techniques.

Acknowledgements

The authors would like to thank the Office of Naval Research, ONR grant N00014-09-1-0069, for funding this research. The opinions expressed in this paper are those of the authors and do not necessarily reflect positions held by the funding agency.

References

Ali, S., and Shapiro, S. 1993. Natural language processing using a propositional semantic network with structured variables. *Minds and machines* 3(4):421–451.

Chklovski, T. 2003. Learner: a system for acquiring commonsense knowledge by analogy. In *K-CAP '03: Proceedings of the 2nd international conference on Knowledge capture*, 4–12. New York, NY, USA: ACM.

de Marneffe, M.-C.; MacCartney, B.; and Manning, C. D. 2006. Generating typed dependency parses from phrase structure parses. In *LREC*.

Fellbaum, C., ed. 1998. *WordNet: An electronic lexical database*. MIT press Cambridge, MA.

Jurafsky, D.; Martin, J.; and Kehler, A. 2000. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. MIT Press.

Kayser, D., and Nouioua, F. 2009. From the textual description of an accident to its causes. *Artificial Intelligence* 173(12-13):1154–1193.

Kipper, K.; Korhonen, A.; Ryant, N.; and Palmer, M. 2006. Extending verbnet with novel verb classes. In *Proceedings of LREC*.

Lenat, D.; Prakash, M.; and Shepherd, M. 1985. Cyc: Using common sense knowledge to overcome brittleness and knowledge acquisition bottlenecks. *AI magazine* 6:65–85.

Liu, H., and Singh, P. 2004. Conceptnet: A practical commonsense reasoning toolkit. *BT Technology Journal* 22.

Liu, H.; Lieberman, H.; and Selker, T. 2006. Goose: A goal-oriented search engine with commonsense. In De Bra, P.; Brusilovsky, P.; and Conejo, R., eds., *Adaptive Hypermedia and Adaptive Web-Based Systems*, volume 2347 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg. 253–263.

Livingston, K., and Riesbeck, C. 2009. Resolving references and identifying existing knowledge in a memory based parser. In *the AAAI 2009 Spring Symposium, Learning by Reading and and Learning to Read, California, USA*.

Mitchell, T. M.; Shinkareva, S. V.; Carlson, A.; Chang, K.-M.; Malave, V. L.; Mason, R. A.; and Just, M. A. 2008. Predicting human brain activity associated with the meanings of nouns. *Science* 320(5880):1191–1195. 10.1126/science.1152876.

Niculescu-Mizil, A., and Caruana, R. 2007. Inductive transfer for bayesian network structure learning. In *Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07)*.

Portet, F.; Reiter, E.; Gatt, A.; Hunter, J.; Sripada, S.; Freer, Y.; and Sykes, C. 2009. Automatic generation of textual summaries from neonatal intensive care data. *Artificial Intelligence* 173(7-8):789–816.

Speer, R.; Havasi, C.; and Lieberman, H. 2008. Analogyspace: Reducing the dimensionality of commonsense knowledge. In *AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence*, 548–553. AAAI Press.

Tsamardinos, I.; Brown, L.; and Aliferis, C. 2006. The max-min hill-climbing bayesian network structure learning algorithm. *Machine Learning* 65(1):31–78.