Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Extraction of Common Physical Properties of Everyday Objects from Structured Sources

Viktor Losing, Julian Eggert

2022

Preprint:

This is an accepted article published in NLPIR '22: Proceedings of the 2022 6th International Conference on Natural Language Processing and Information Retrieval. The final authenticated version is available online at: https://doi.org/10.1145/3582768.3582772

© Julian Eggert 2022. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in the ACM DL conference proceedings, http://dx.doi.org/10.1145/3582768.3582772.

Extraction of Common Physical Properties of Everyday Objects from Structured Sources

Viktor Losing
viktor.losing@honda-ri.de
Honda Research Institute Europe GmbH
Offenbach am Main, Germany

Julian Eggert
julian.eggert@honda-ri.de
Honda Research Institute Europe GmbH
Offenbach am Main, Germany

ABSTRACT

Commonsense knowledge is essential for the reasoning of AI systems, particularly in the context of action planning for robots. The focus of this paper is on common-sense object properties, which are especially useful to restrict the search space of planning algorithms. Popular sources for such knowledge are commonsense knowledge bases that provide the information in a structured form. However, the utility of the provided object-property pairs is limited as they can be simply incorrect, subjective, unspecific, or relate only to a narrow context. In this paper, we suggest a methodology to create a highly accurate dataset of object properties that are related to common physical attributes. The approach is based on filtering non-physical properties within commonsense knowledge bases and improving the accuracy of the remaining object-property pairs based on supervised machine learning using annotated data. Thereby, we evaluate different types of features and models and significantly increase the correctness of object-property pairs compared to the original sources.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**; **Neural networks**; *Planning and scheduling*; *Knowledge representation and reasoning*.

KEYWORDS

transformer model, knowledge bases, neural networks

ACM Reference Format:
Viktor Losing and Julian Eggert. 2022. Extraction of Common Physical Properties of Everyday Objects from Structured Sources. In *2022 6th International Conference on Natural Language Processing and Information Retrieval (NLPIR 2022), December 16–18, 2022, Bangkok, Thailand.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3582768.3582772

1 INTRODUCTION

Commonsense knowledge (CSK) is crucial for performing everyday tasks and humans acquire them throughout their life, mostly in an effortless manner. Transferring such knowledge to Artificial Intelligence (AI) systems is a necessity for them to be robustly applied in a variety of tasks and environments. One way to transfer such knowledge is to collect it in a structured form, often in large knowledge graphs and to provide it to the system as a foundation for the reasoning process. The creation of such knowledge graphs and their application in AI systems is a long endeavor initiated already several decades ago [8, 9, 15].

Recently, advances in Natural Language Processing (NLP) and automatic acquisition of commonsense assertions resulted in various large knowledge bases that have been mostly used in language-related tasks such as question answering, conversational systems, and storytelling [4, 5, 17, 20], but are also applied in action planning for robots [1, 2, 12]. We are interested in the latter use case and are particularly interested in viable and distinctive properties of everyday objects, as these are crucial to restrict the search space of planning systems.

Even though common object properties are contained in various modern commonsense knowledge bases (CSKB) the majority of them are irrelevant for planning systems. This is due to the fact that most data is automatically extracted where all adjectives in the text that are related to the object are potential property candidates. Therefore, the sources contain irrelevant or close to invalid / incorrect properties that simply do not apply to a given object, also subjective properties that state a preference such as "best", "favorite", "beautiful", "hated". Also adjectives that are related to the current context, or are comparative, e.g., "lowest", "above", "first", "ten", "larger", "sweeter" end up in such databases.

We are mainly interested in properties that describe the common physical attributes of an object that can be perceived by our senses. Some examples include the color ("red apple") or shape "round apple", its texture "smooth apple", softness "hard apple", temperature "hot apple", weight "light apple", or its taste "sweet apple" and scent "smelly apple". We are furthermore interested in properties that describe a common state of the object and affect its physical attributes such as "cut apple", "squashed apple", "boiled apple", and "canned apple".

This paper proposes a straightforward approach to extracting such physical properties for a set of everyday objects.Concretely, we define a set of everyday objects based on an extension of the Microsoft COCO dataset [7] and use the properties extracted from five popular CSKBs as starting point. We filter irrelevant and non-physical adjectives from the candidates and construct an annotated dataset of valid and invalid object-property pairs (OP pairs) based on a sample of the remaining data. Subsequently, we train and evaluate a different set of features and models for this task, thereby significantly increasing the correctness of the OP pairs. The final dataset is obtained by filtering the invalid object properties from the unlabeled data using the best-performing model.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
NLPIR 2022, December 16–18, 2022, Bangkok, Thailand
© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9762-9/22/12…$15.00
https://doi.org/10.1145/3582768.3582772

2 DATASET CONSTRUCTION

The dataset is constructed in two main steps. First, we extract for a set of everyday objects all available properties from multiple external databases and remove non-physical properties. Then we improve the accuracy of the remaining OP pairs by annotating a sample of the data into valid and invalid pairs and applying various machine learning models to the task. The OP pairs that are classified as valid are collected in the final dataset.

2.1 Set of Everyday Objects

We use the object categories from the Microsoft COCO dataset and add additional 33 common household objects to define our set of everyday objects. The 113 object categories are manually mapped to WordNet [11] synsets. WordNet is a large lexical database of English words that are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. This enables us to grow our set of object names with their synonyms assigned to the synsets. Altogether, the constructed dataset contains 218 different object names.

2.2 External Sources

We use five popular common-sense knowledge bases that already provide OP pairs in a structured form. The majority of the pairs were automatically extracted from a large text corpus and a small part was collected via crowd-sourcing. In the following, we describe the utilized CSKBs.

2.2.1 ConceptNet. ConceptNet [16] is a knowledge graph that connects words and phrases of natural language with weighted labeled edges. It is mainly based on crowd-sourcing or expert-created resources. The type of common-sense knowledge is distinguished by the limited set of labeled edges, often called relations. In our case, we solely focus on the "Has-Property" relation. The weight of the edge can be interpreted as a rough confidence score calculated based on the frequency of the object-property occurrence and the type of original source it was extracted from.

2.2.2 WebChild. WebChild[18] is a large collection of common-sense knowledge, that is automatically extracted from Web contents. WebChild contains triplets that connect nouns with adjectives via 19 different relations. We focus solely on the "hasProperty" relation. For each object property, a frequency is provided as confidence for its validity.

2.2.3 Quasimodo. Quasimodo [14] extracts the majority of its information from the Web by tapping into search-engine query logs and QA forums like Reddit and Quora. Quasimodo extracts typical properties and predicates that are associated with certain objects and concepts. We use for our purposes only the information that belongs to the predicate "has_property". Quasimodo calculates from the frequency of found assertions two confidence-related scores termed "typicality" and "saliency".

2.2.4 TransOMCS. TransOMCS[19] extracts commonsense knowledge from a large collection of linguistic graphs. Thereby, it uses the established relation set of ConceptNet, but provides more than three times as many assertions. As for ConceptNet we use solely the "Has-Property" relation. Each assertion is coupled with a frequency-based confidence score.

2.2.5 Ascent++. Ascent++[13] is a recently published common-sense knowledge graph that is extracted from a large English text corpus. Additionally, the common triplet-based information also provides composite concepts and so-called facets. We rely on the variant of Ascent++ that uses the established ConceptNet relations and limit the extraction on the "Has-Property" relation. Similar to Quasimodo two scores named "typicality" and "saliency" are provided. However, in contrast to Quasimodo, the "typicality" score is not based on the frequency alone but is predicted by a regression model.

2.3 Filtering

Starting with all available properties for our object names within the five CSKBs, we perform several filtering steps. First, we remove all pairs where the extracted property is not represented as an adjective within WordNet. This can be seen as a basic sanity check of the extraction results and removes terms that are either no adjectives or typos of those. This results in a set of 94476 of OP pairs with 8363 unique properties. The majority of the 8363 properties, as already mentioned, are irrelevant to our purpose as they do not describe the physical attributes of the objects. Therefore, we filter those manually and end up at our set of 1395 relevant properties, which results in a set of 21345 relevant OP pairs.

Figure 1 shows the contribution of the different sources to the data set before and after filtering. Most OP pairs are found within WebChild, followed by TransOMCS, Ascent++, and Quasimodo. ConceptNet provides only a negligible fraction of the object properties.

For most sources, the filtering removes $\sim$ 75% of the data. In the case of Ascent++ and ConceptNet, $\sim$ 60% and 35% of the data are filtered which indicates higher quality and more practical relevance of the provided information.

2.4 Annotation

We manually annotate a random sample of 1200 pairs of our set of 21345 relevant OP pairs. Hereby, we follow the approach in [13] and distinguish the level of correctness into four levels:

(1) Always/Often - the object-property pair is always/often true
(2) Sometimes/Likely - it is sometimes or likely true
(3) Far-fetched/Never - it is far-fetched at best or never true
(4) Invalid - it is invalid and makes no sense

The annotated data serves the purpose to assess the initial correctness of the relevant OP pairs, but also to building machine learning models that allow filtering of invalid pairs in order to improve the correctness. This means that we reduce this to a binary classification problem where we label the OP pairs that are assigned to the first two levels as valid and those assigned to the latter two levels as invalid.

Figure 2 depicts the amount of valid and invalid pairs within the annotated data grouped by the corresponding source. Even though our sample is relatively small, we get a rough picture of the initial correctness of the extracted information. ConceptNet has the highest accuracy but represents only a small part of the dataset. Among

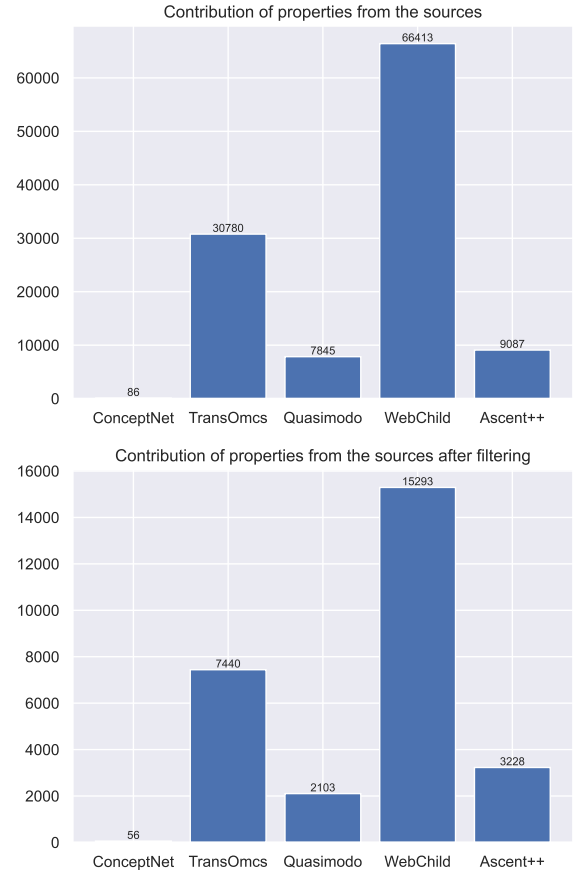

Figure 1: Contribution of OP pairs from the different sources before (upper) and after (lower) filtering.

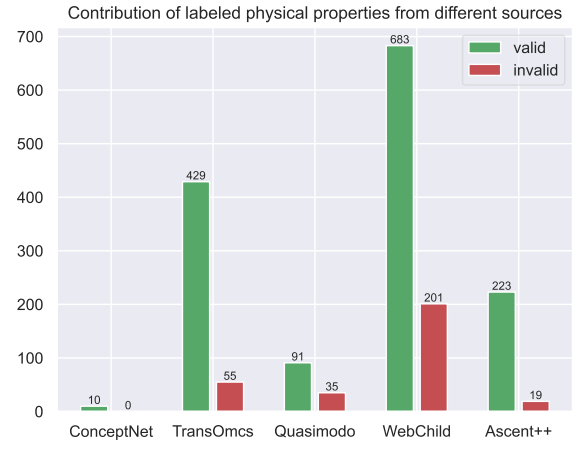

Figure 2: Contribution of valid and invalid OP pairs from the different sources within the annotated data.

the larger sources, Ascent++ has the highest accuracy within our sample and WebChild the lowest. These estimates are in line with the figures reported in [13]. The annotated dataset is imbalanced with 78% valid and 22% invalid properties, which indicates that the majority of the OP pairs in the CSKBs is correct, but far from being perfect [1].

3 EVALUATION

We evaluate various models with different sets of features. The simplest approach is a logistic regression model using only the confidence scores of the original CSKBs, whereas the most sophisticated is the pre-trained language model BERT (base-uncased) [3] fine-tuned on the annotated data and enriched with additional input. Table 1 gives an overview of all considered approaches. The CSKBs confidences are the confidence scores of the five CSKB for a specific object-property pair. The respective score is set to 0 if a pair is not represented in a CSKB. In the case of the Word2Vec [10] embeddings, we concatenate the embeddings of the object and the property into a vector of 600 dimensions. The embeddings of BERT are obtained by feeding the template sentence "This {object} is {property}." into the model. We use the BERT embeddings also as input into the logistic regression model. The difference to the BERT model is that only the weights of the classification layer are adapted, whereas fine-tuning of BERT implies the modification of all previous layers as well.

We use 70% of the annotated data for training, 10% for validation and the rest for testing. All experiments are repeated 30 times, thereby randomly sampling a new train and test set, to estimate the variance of the model performance.

We mainly use the Area Under the Curve (AUC) of Receiver Operating Curves (ROC) as an evaluation metric, since our data set is imbalanced (78% valid vs. 22% invalid). The AUC has furthermore the advantage that it summarizes the model performance using all viable detection thresholds. This is for instance not the case for the popular F1 score that provides only a point estimate with a specific detection threshold.

3.1 Results

Figure 3 shows the ROCs of different logistic regression models. Looking at the *logreg_conf* model, it can be seen that the confidences of the source databases have some predictive information as the model performs better than chance. However, using the learned representation of the object and property words leads to significantly higher AUC scores, as can be seen for the models *logreg_w2v* and *logreg_bert*. As expected, the BERT embeddings prove to be superior in comparison to Word2Vec as it has been repeatedly shown in literature [3, 6]. Nonetheless, the best results are achieved when all features are combined, indicating that the confidence scores and Word2Vec embeddings do provide some additional benefit.

The ROCs of the fine-tuned language models are depicted in Figure 4. The models perform similarly indicating that the majority of the performance is due to the fine-tuning of the BERT embeddings.

[1] It is worthwhile to remind the reader that these estimates are inferred from already filtered data. Therefore, it is likely that the unfiltered data has an even larger proportion of invalid pairs.

Table 1: Evaluated approaches

Abbreviation	Approach	Features
logreg_conf	Logistic regression	CSKBs confidences
logreg_w2v	Logistic regression	Word2vec embeddings
logreg_bert	Logistic regression	Bert embeddings
logreg_all	Logistic regression	CSKBs confidences, Word2Vec embeddings, BERT embeddings
bert	Fine-tuned BERT model	BERT embeddings
bert_conf	Fine-tuned BERT model	BERT embeddings, CSKBs confidences
bert_w2v	Fine-tuned BERT model	BERT embeddings, Word2Vec embeddings
bert_all	Fine-tuned BERT model	CSKBs confidences, Word2Vec embeddings, BERT embeddings

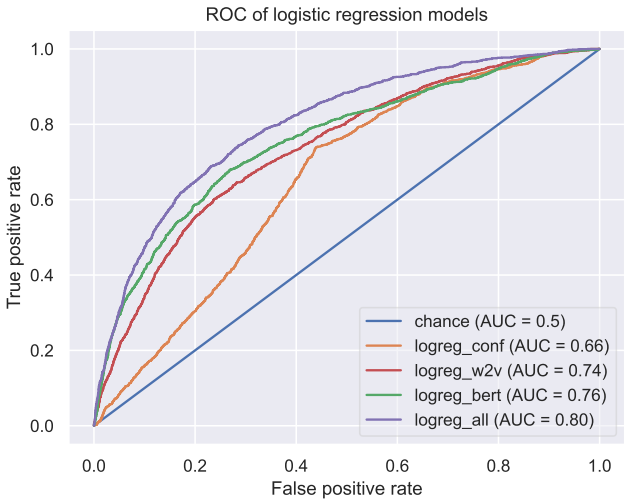

Figure 3: Average ROC curves of all logistic regression models based on 30 repetitions. All models perform distinctly better than chance. The model *logreg_all* that uses as feature vector the confidence scores, Word2Vec, and BERT embeddings delivers the best results.

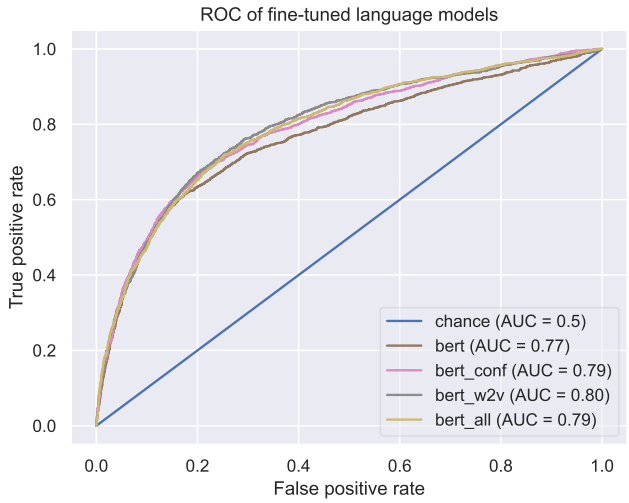

Figure 4: Average ROC curves of the fine-tuned language models based on 30 repetitions.

However, incorporating additional input such as the Word2Vec embeddings or the confidence scores of the CSKBs slightly increases the performance further.

3.1.1 Learning Curve. The required amount of annotated data for specific performance is a critical question in the context of model selection. Figure 5 provides an answer by depicting the learning curve of the most interesting approaches. The model that uses only the confidences of the source database (*logreg_conf*) converges already after around 100 examples, but yields only an average AUC score of 0.68, which nonetheless is better than chance.

The model *logreg_all* has initially a steeper learning curve than *bert_all* but both provide a similar performance when all training data is used. These approaches appear to be not yet saturated suggesting that higher performance can be achieved with additional training data. Comparing the logistic regression models with the fine-tuned BERT models, we can conclude that the BERT embeddings provide a useful representation. However, we do not observe a clear advantage of fine-tuning them for this specific task. Furthermore, the fine-tuning approach has a higher computational complexity during training.

We assume that the adaptation of the classification layer using the BERT embeddings as input gives the logistic regression model enough degrees of freedom to perform well on our task and that larger amounts of annotated data are required to gain an advantage with adapting the whole representation through fine-tuning.

3.2 Final Dataset

To create the final dataset of OP pairs, we need to set a detection threshold and apply the model to the remaining unlabeled OP pairs. The pairs that are classified as invalid are then removed from the final data set.

We select the threshold that maximizes the F1 score for the detection of invalid examples on the annotated data. Table 2 provides an overview of the corresponding results. Maximizing the F1 score is one way to pick a balance between the amount of remaining OP pairs after filtering and the accuracy within this dataset. The more pairs are classified as invalid and removed from the data the higher the accuracy, as only highly confident pairs remain in the set.

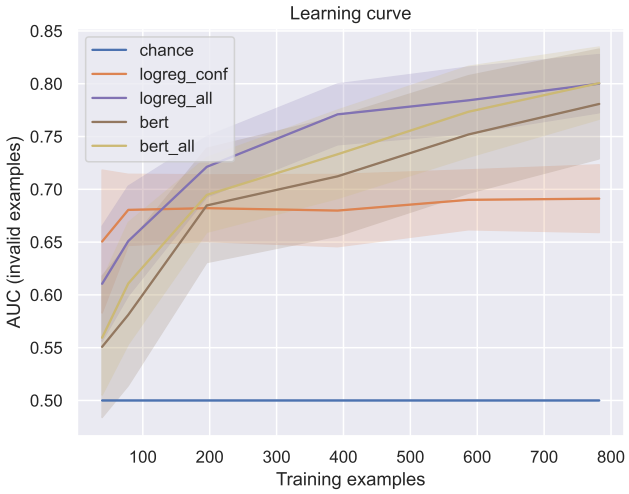

Figure 5: Average learning curves of some selected approaches after 30 repetitions. Fine-tuning the BERT model has no clear advantage over simply using its pre-trained embeddings as input.

Table 2: Remaining OP pairs and their accuracy after filtering invalid examples by selecting the threshold that maximizes the F1 score. The test set before filtering consists of 245 OP pairs with an average accuracy of 0.782 ± 0.034

Model	F1	Accuracy	#OP pairs
logreg_conf	0.468±0.035	0.842±0.069	110±19
logreg_w2v	0.508±0.038	0.850±0.036	160±23
logreg_bert	0.533±0.039	0.855±0.038	167±18
logreg_all	**0.572±0.034**	**0.874±0.034**	167±19
bert	0.550±0.045	0.860±0.029	188±11
bert_conf	0.561±0.061	0.862±0.030	**193±14**
bert_w2v	0.559±0.046	0.861±0.027	**193±12**
bert_all	0.569±0.054	0.863±0.027	191±10

All models can improve the initial accuracy of 0.782 quite significantly. The highest F1 score and accuracy is achieved by *logreg_all*. The BERT models preserve a larger amount of OP pairs compared to *logreg_all* at the cost of lower accuracy. In other words, the BERT models produce a lower amount of false positives but more false negatives.

Applying the model *logreg_all* on our set of relevant OP pairs leads to our final dataset. Thereby, the amount of OP pairs is reduced from 21345 to 14422, corresponding to a reduction of 32%.

4 CONCLUSION

We proposed a methodology to extract physical attributes for a set of everyday objects from multiple CSKBs, fusing the sources and filtering irrelevant non-physical properties. We increased the correctness of the remaining OP pairs by creating an annotated data set of valid and invalid pairs and applying machine learning models. While the confidence scores of the CSKBs provide some predictability for this task, we drastically improved the performance with the embeddings of the pre-trained BERT. The best-performing model improves the accuracy of the OP pairs from .78 to .87 with the cost of reducing the number of extracted OP pairs by 32%.

In the future, we want to increase our annotated dataset via crowdsourcing and provide it to the public. Furthermore, we aim to analyze the embeddings of larger pre-trained language models.

REFERENCES

[1] Michael Beetz, Daniel Beßler, Andrei Haidu, Mihai Pomarlan, Asil Kaan Bozcuoğlu, and Georg Bartels. 2018. Know rob 2.0—a 2nd generation knowledge processing framework for cognition-enabled robotic agents. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 512–519.

[2] Christopher J Conti, Aparna S Varde, and Weitian Wang. 2020. Robot action planning by commonsense knowledge in human-robot collaborative tasks. In *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[4] Deepanway Ghosal, Navonil Majumder, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. 2020. Cosmic: Commonsense knowledge for emotion identification in conversations. *arXiv preprint arXiv:2010.02795* (2020).

[5] Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6473–6480.

[6] Andrey Konstantinov, Vadim Moshkin, and Nadezhda Yarushkina. 2020. Approach to the use of language models BERT and Word2vec in sentiment analysis of social network texts. In *International Scientific and Practical Conference in Control Engineering and Decision Making*. Springer, 462–473.

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.

[8] Cynthia Matuszek, Michael Witbrock, John Cabral, and John DeOliveira. 2006. An introduction to the syntax and content of Cyc. *UMBC Computer Science and Electrical Engineering Department Collection* (2006).

[9] John McCarthy et al. 1960. *Programs with common sense*. RLE and MIT computation center Cambridge, MA, USA.

[10] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. http://arxiv.org/abs/1301.3781

[11] George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

[12] Tiago Mota and Mohan Sridharan. 2019. Commonsense Reasoning and Knowledge Acquisition to Guide Deep Learning on Robots.. In *Robotics: science and systems*.

[13] Tuan-Phong Nguyen, Simon Razniewski, Julien Romero, and Gerhard Weikum. 2021. Refined commonsense knowledge from large-scale web contents. *arXiv preprint arXiv:2112.04596* (2021).

[14] Julien Romero, Simon Razniewski, Koninika Pal, Jeff Z. Pan, Archit Sakhadeo, and Gerhard Weikum. 2019. Commonsense properties from query logs and question answering forums. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1411–1420.

[15] Edward H Shortliffe, Bruce G Buchanan, and Edward A Feigenbaum. 1979. Knowledge engineering for medical decision making: A review of computer-based clinical decision aids. *Proc. IEEE* 67, 9 (1979), 1207–1224.

[16] Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

[17] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937* (2018).

[18] Niket Tandon, Gerard De Melo, and Gerhard Weikum. 2017. Webchild 2.0: Fine-grained commonsense knowledge distillation. In *Proceedings of ACL 2017, System Demonstrations*. 115–120.

[19] Hongming Zhang, Daniel Khashabi, Yangqiu Song, and Dan Roth. 2020. TransOMCS: From Linguistic Graphs to Commonsense Knowledge. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI) 2020*.

[20] Wanjun Zhong, Duyu Tang, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2019. Improving question answering by commonsense-based pre-training. In *CCF International Conference on Natural Language Processing and Chinese Computing*.