Are We Modeling the Task or the Annotator? An Investigation of Annotator Bias in Natural Language Understanding Datasets

Mor Geva
Tel Aviv University,
Allen Institute for AI
morgeva@mail.tau.ac.il

Yoav Goldberg
Bar-Ilan University,
Allen Institute for AI
yoav.goldberg@gmail.com

Jonathan Berant
Tel Aviv University,
Allen Institute for AI
joberant@cs.tau.ac.il

Abstract

Crowdsourcing has been the prevalent paradigm for creating natural language understanding datasets in recent years. A common crowdsourcing practice is to recruit a small number of high-quality workers, and have them massively generate examples. Having only a few workers generate the majority of examples raises concerns about data diversity, especially when workers freely generate sentences. In this paper, we perform a series of experiments showing these concerns are evident in three recent NLP datasets. We show that model performance improves when training with annotator identifiers as features, and that models are able to recognize the most productive annotators. Moreover, we show that often models do not generalize well to examples from annotators that did not contribute to the training set. Our findings suggest that annotator bias should be monitored during dataset creation, and that test set annotators should be disjoint from training set annotators.

1 Introduction

Generating large datasets has become one of the main drivers of progress in natural language understanding (NLU). The prevalent method for creating new datasets is through crowdsourcing, where examples are generated by workers (Zaidan and Callison-Burch, 2011; Richardson et al., 2013; Bowman et al., 2015; Rajpurkar et al., 2016; Trischler et al., 2017). A common recent practice is to choose a small group of workers who produce high-quality annotations, and massively generate examples using these workers.

Having only a few workers annotate the majority of dataset examples raises concerns about data diversity and the ability of models to generalize, especially when the crowdsourcing task is to generate free text. If an annotator consistently uses language patterns that correlate with the labels, a neural model can pick up on those, which can lead to an over-estimation of model performance.

In this paper, we continue recent efforts to understand biases that are introduced during the process of data creation (Levy et al., 2015; Schwartz et al., 2017; Gururangan et al., 2018; Glockner et al., 2018; Poliak et al., 2018; Tsuchiya, 2018; Aharoni and Goldberg, 2018; Paun et al., 2018). We investigate this form of bias, termed *annotator bias*, and perform multiple experiments over three recent NLU datasets: MNLI (Williams et al., 2018), OpenBookQA. (Mihaylov et al., 2018), and CommonsenseQA (Talmor et al., 2019).

First, we establish that annotator information improves model performance by supplying annotator IDs as part of the input features. Second, we show that models are able to recognize annotators that generated many examples, illustrating that annotator information is captured by the model. Last, we test whether models generalize to annotators that were not seen at training time. We observe that often generalization to new annotators fails, and that augmenting the training set with a small number of examples from these annotators substantially increases performance.

Taken together, our experiments show that annotator bias exists in current NLU datasets, which can lead to problems in model generalization to new users. Hence, we propose that annotator bias should be monitored at data collection time and to tackle it by having the test set include examples from a disjoint set of annotators.

2 Crowdsourcing Practice

Crowdsourcing has become the prominent paradigm for creating NLP datasets (Callison-Burch et al., 2015; Sabou et al., 2014). It has been used for various NLU tasks, including Question Answering (Rajpurkar et al., 2018; Mihaylov

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 1161–1166,
Hong Kong, China, November 3–7, 2019.

et al., 2018; Dua et al., 2019), commonsense and visual reasoning (Talmor et al., 2019; Zellers et al., 2019; Suhr et al., 2018), and Natural Language Inference (NLI) (Williams et al., 2018).

In a typical process, annotators are recruited and screened (Sabou et al., 2014), often resulting in a small group that creates most of the dataset examples. Mihaylov et al. (2018) recruited a few dozens of qualified workers that wrote 5,957 questions. Suhr et al. (2018) recruited 99 workers for a sentence writing task, who created more than 100,000 examples. Williams et al. (2018) recruited 387 workers for writing over 400,000 sentences, while Krishna et al. (2017) had 33,000 workers contributing 1.7 million examples.

These examples demonstrate that datasets are often constructed using a small number of annotators, approximately 1 annotator per 10^2–10^3 examples. Furthermore, (see Section 3), the annotator distribution is skewed with a few annotators creating the vast majority of the dataset. In tasks that involve creative language writing, this may have implications on data diversity, and lead to an over-estimation of model performance.

3 Experimental Setup

We focus on crowdsourcing tasks where workers produce full-length sentences. We first describe the datasets we test our hypothesis on, and then provide details on the model and training.

Datasets We consider recent NLU datasets, for which the annotator IDs are available.

- **MNLI** (matched) (Williams et al., 2018): A NLI dataset. Each example was created by introducing an annotator with a premise sentence and asking her to write a hypothesis sentence that is either entailed, contradicted or is neutral with respect to the premise.
- **OpenBookQA** (Mihaylov et al., 2018): A multiple-choice question answering dataset, focusing on multi-hop reasoning. Each question and its answer distractors were written by a worker, based on a given scientific fact.
- **CommonsenseQA** (Talmor et al., 2019): A multiple-choice question answering dataset, focused on commonsense knowledge. Questions were written by crowdworkers, who try to bridge between two concepts extracted from ConceptNet (Speer et al., 2017)

	# examples	# annotators
MNLI (matched)	402,517	380
OpenBookQA	5,457	84
CommonsenseQA	11,096	132

Table 1: Statistics for datasets used in our experiments.

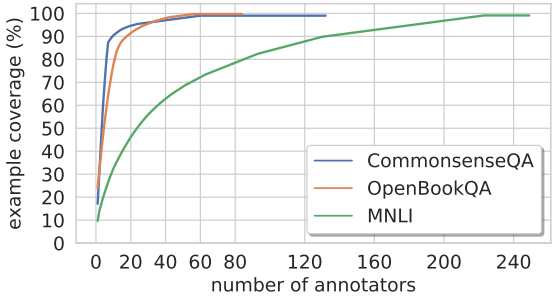

Figure 1: Proportion of examples covered by number of annotators (sorted by number of annotations).

Table 1 summarizes the size and number of annotators who worked on each dataset. Figure 1 shows the fraction of examples covered by the number of annotators, sorted by the number of examples they annotated. In all datasets, and specifically in OpenBookQA and CommonsenseQA, most of the examples were generated by a small number of annotators.

Model We use the pretrained BERT-base (Devlin et al., 2018), a strong model obtaining close to state-of-the-art performance on all three datasets. We add a single linear layer over BERT outputs and apply the same fine-tuning procedure in all experiments: fine-tuning for 3 epochs, using batch size 10, learning rate 2×10^{-5}, and maximum sequence length of 128 word pieces.

4 Experiments and Results

We now describe a series of experiments for quantifying annotator bias, aiming to answer the following questions: 1) Do models perform better when exposed to the annotator ID? 2) Can models detect the annotators from their generated examples? 3) Do models generalize across annotators?

The utility of annotator information Our first experiment aims to determine whether models perform better given *perfect* information on the annotator ID. To this end, we follow the standard way of feeding input to BERT (Devlin et al., 2018) and concatenate the annotator ID as an additional feature to every example in every dataset. Formally, we replace every example ($x =$

$(w_1, ..., w_{|x|}), y)$ created by annotator z, with the example $((z, w_1, ..., w_{|x|}), y)$, where z is a textual unique annotator identifier, x is the input sequence and y is the gold label.

We compare performance on the original datasets and their new version. Adding the annotator ID improves model performance across all datasets (Table 2), showing that perfect annotator information is useful for prediction, and there is incentive for the model to capture this information.

	Without ID	With ID	p-value
OpenBookQA	52.2	56.4	$1.83e^{-2}$
CommonsenseQA	53.6	55.3	$11.98e^{-2}$
MNLI	82.9	84.5	$5.13e^{-7}$

Table 2: Model development performance, after training with/without annotator IDs as additional inputs. Following Dror et al. (2018), p-values were calculated using the McNemar's test (McNemar, 1947) for MNLI and the Bootstrap test (Berg-Kirkpatrick et al., 2012) for OpenBookQA and CommonsenseQA.

Annotator recognition Perfect annotator information improves performance, but it is possible that a model can recognize the annotators from the input sequence only, even without being exposed to the annotator ID explicitly. In the next experiment, we investigate the ability of models to recognize the annotators from the input.

To this end, we fine-tune BERT-base to predict annotator IDs from examples. We limit the task to 6 labels of the top-5 most productive annotators of each dataset and an OTHER label for all other annotators. Formally, we replace every example (x, y) created by annotator z, with the example $(x, \bar{z})$, where $\bar{z} = z$ if z is in the top-5 annotators and $\bar{z} =$ OTHER otherwise.

Figure 2 shows the F1-score for the top-5 annotators of every dataset (y-axis), and the fraction of dataset examples created by each annotator (x-axis). Overall, annotators who write many examples are recognized better by the model: The model struggles to recognize MNLI annotators with F1 scores below 0.5, and excels at recognizing annotators from CommonsenseQA with scores between 0.76–0.91. For the top annotator of OpenBookQA, who created 24% of the dataset examples, the model obtains a high F1 score of 0.8. To conclude the first two experiments, annotator ID information is useful for the downstream task, and also can be predicted with high probability from the input for a large fraction of the examples.

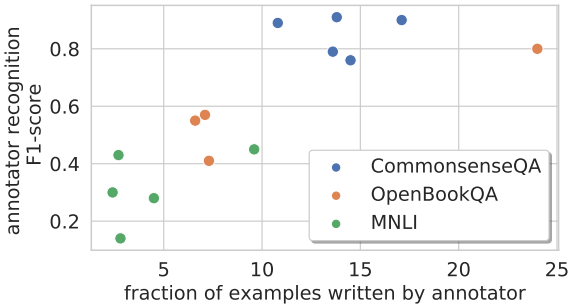

Figure 2: Annotator recognition F1-scores for the top-5 annotators of each dataset. For OpenBookQA only 4 data points are plotted, as the second annotator is not in the original development set.

Generalization across annotators In our final experiment, we examine whether trained models generalize to new annotators. To address this question, we re-split each dataset, creating a training and development set with disjoint annotators: Given a dataset with example set $\mathcal{S}$, we denote by $\mathcal{S}_z \subset \mathcal{S}$ the subset of examples created by annotator z. Similarly, for a set of annotators $\mathcal{Z}$, let $\mathcal{S}_\mathcal{Z} = \bigcup_{z \in \mathcal{Z}} \mathcal{S}_z$. We rank annotators by the number of examples they generated, and for each dataset $\mathcal{S}$, we create two types of data splits. For each annotator z in the top-5 annotators, we create a *single-annotator* data split with $\mathcal{S} \backslash \mathcal{S}_z$ and $\mathcal{S}_z$ as the train and development sets, respectively. Namely, we consider the examples created by annotator z as the development set, while using all other examples for training. In addition, we pick 5 sets of 5 annotators, who annotated a small number of examples, and for each such set $\mathcal{Z}$, we create a *multi-annotator* split with $\mathcal{S} \backslash \mathcal{S}_\mathcal{Z}$ and $\mathcal{S}_\mathcal{Z}$ as the train and development sets, respectively. Namely, we consider the examples created by the 5 annotators $\mathcal{Z}$ as the development set, while using all other examples for training. Overall, there are 5 single-annotator splits and 5 multi-annotator splits for each dataset.

We fine-tune BERT-base and evaluate it on the development set, and compare the results to a random data split of identical size. We repeat every experiment 3 times, except for multi-annotator experiments on OpenBookQA and CommonsenseQA which we repeat 9 times due to high variance. Table 3 shows the mean and standard deviation of performance difference (p.d.) between each annotator(s) split and its corresponding ran-

dom split, where negative numbers indicate that performance on the annotator split was lower.

Our clearest finding is that in OPENBOOKQA performance on the the multi-annotator split is dramatically lower than on a random split in all 5 annotator sets, where performance drops by up to 23 accuracy points. This shows that the model does not generalize to examples generated by unseen annotators. In the other datasets, results on the multi-annotator split are more varied, where performance drops in roughly half the cases, sometimes substantially – up to 10 accuracy points in COMMONSENSEQA and 5 in MNLI.

In the single-annotator splits, in roughly half the cases performance on the annotator split was lower than the random split. However, measuring p.d. only for single annotators might be misleading, because specific annotators vary in the difficulty of examples they produce. Thus, running a model on a new annotator that produces easy examples will not result in decreased performance. Next, we propose a more fine-grained experiment that controls for these two confounding factors.

COMMONSENSEQA-single		COMMONSENSEQA-multi	
4.2 ± 0.7	17.1%	-9.5 ± 8.3	0.9%
7.7 ± 1.9	14.5%	6.5 ± 7.0	0.6%
-2.8 ± 1.3	13.8%	-6.1 ± 8.5	0.5%
-3.8 ± 0.9	13.6%	1.6 ± 10.8	0.4%
1.6 ± 2.7	10.8%	1.8 ± 10.5	0.4%
OPENBOOKQA-single		OPENBOOKQA-multi	
-0.9 ± 2.7	24%	-14.7 ± 6.2	2.4%
-13.5 ± 1.7	7.8%	-19.4 ± 8.5	1.7%
-5.8 ± 0.7	7.3%	-12.4 ± 5.5	1.2%
8.2 ± 5.2	7.1%	-13.7 ± 8.5	1%
3.1 ± 1.1	6.6%	-23.3 ± 7.8	0.8%
MNLI-single		MNLI-multi	
-2.5 ± 0.5	9.6%	2.5 ± 0.8	1.8%
-3.0 ± 0.6	4.5%	-1.1 ± 0.9	1.5%
2.9 ± 0.2	2.8%	-4.6 ± 0.8	1.5%
0.8 ± 0.7	2.7%	-1.5 ± 0.2	1.5%
4.6 ± 0.2	2.4%	0.5 ± 0.2	1.5%

Table 3: Performance difference (p.d.) between single- and multi- annotator splits and random splits of identical size. Each cell shows the p.d. mean and standard deviation, as well as the development set relative size.

Separating annotator bias from annotator difficulty To mitigate the effect of annotator difficulty, we perform an augmentation experiment. Assume a development set $\mathcal{S}_{\text{dev}}$, for which performance is low. Our hypothesis is that if $\mathcal{S}_{\text{dev}}$ is inherently difficult, then moving a small portion of examples from $\mathcal{S}_{\text{dev}}$ to the training set $\mathcal{S}_{\text{train}}$ should not change performance on $\mathcal{S}_{\text{dev}}$ substantially. However, if performance on $\mathcal{S}_{\text{dev}}$ is low due to annotator bias, then moving examples to $\mathcal{S}_{\text{train}}$ would expose the model to the annotator and performance should go up.

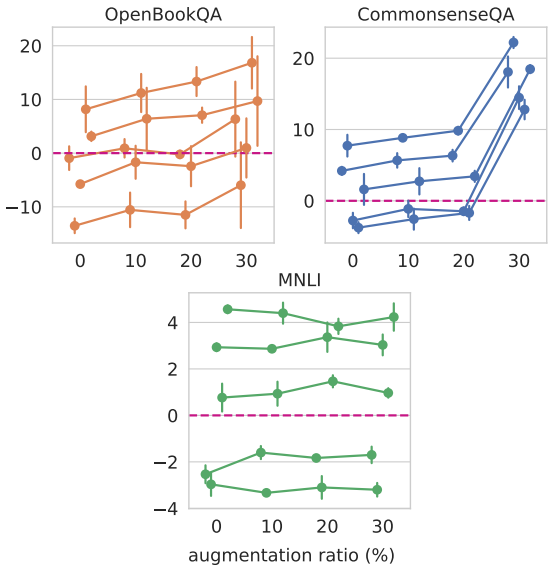

Figure 3: Performance difference between single-annotator splits and random splits of identical size. The x-axis indicates the fraction of examples taken from the development set to augment the training set.

For every single-annotator data split $\mathcal{S}_{\text{train}}$, $\mathcal{S}_{\text{dev}}$, we perform a series of augmentation experiments, where we move a random fraction of k examples from $\mathcal{S}_{\text{dev}}$ to $\mathcal{S}_{\text{train}}$, for $k = 0.1, 0.2, 0.3$. We keep the size of $\mathcal{S}_{\text{train}}$ constant by randomly removing examples from it before augmentation. We repeat experiments multiple times as before, and report the p.d mean and standard deviation in Figure 3.

Results for both COMMONSENSEQA and OPENBOOKQA show a rapid increase of 10-20 accuracy point for all top-5 annotators, given only a small number of their generated examples. This shows that the examples generated by these annotators are not inherently difficult, and that the model can substantially improve performance by being exposed to the language that the annotators generate. Conversely, performance changes are marginal for MNLI, suggesting generalization patterns are mostly due to example difficulty. The different results for MNLI compared to those observed for OPENBOOKQA and COMMONSENSEQA may be attributed to the less-skewed annotator distribution and large number of examples in MNLI (see Figure 1 and Table 1), which make the model more robust to small perturbations in the data distribution.

5 Discussion and Conclusions

This study set out to investigate whether prevalent crowdsourcing practices for building NLU datasets introduce an annotator bias in the data that leads to an over-estimation of model performance. We established that perfect annotator information can improve model performance, and that the language generated by annotators often reveals their identity. Moreover, we tested the ability of models to generalize to unseen annotators in three recent NLU datasets, and found that in two of these datasets annotator bias is evident. These findings may be explained by the annotator distributions and the size of these datasets. Skewed annotator distributions with only a few annotators creating the vast majority of examples are more prone to biases.

Our results suggest that annotator bias should be monitored in crowdsourcing tasks involving free text generation by annotators. This can be done by testing model performance on new annotators during data collection. Moreover, to tackle annotator bias, we propose that training set annotators should be separated from test-set annotators.

Acknowledgments

This research was partially supported by The Israel Science Foundation grant 942/16, The Blavatnik Computer Science Research Fund and The Yandex Initiative for Machine Learning. We thank Sam Bowman from NYU University and Alon Talmor from Tel Aviv University for providing us the annotation information of the MNLI and COMMONSENSEQA datasets. This work was completed in partial fulfillment for the Ph.D degree of the first author.

References

Roee Aharoni and Yoav Goldberg. 2018. Split and rephrase: Better evaluation and stronger baselines. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 719–724. Association for Computational Linguistics.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An empirical investigation of statistical significance in nlp. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 995–1005. Association for Computational Linguistics.

S. Bowman, G. Angeli, C. Potts, and C. D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Chris Callison-Burch, Lyle Ungar, and Ellie Pavlick. 2015. Crowdsourcing for NLP. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorial Abstracts*, pages 2–3.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhikers guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proc. of NAACL*.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

S. Gururangan, S. Swayamdipta, O. Levy, R. Schwartz, S. R. Bowman, and N. A. Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual Genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. Do supervised distributional methods really learn lexical inference relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *In proceedings of the 2018 Conference*

on Empirical Methods in Natural Language Processing (EMNLP).

Silviu Paun, Bob Carpenter, Jon Chamberlain, Dirk Hovy, Udo Kruschwitz, and Massimo Poesio. 2018. Comparing bayesian models of annotation. *Transactions of the Association for Computational Linguistics*, 6:571–585.

A. Poliak, J. Naradowsky, A. Haldar, R. Rudinger, and B. V. Durme. 2018. Hypothesis only baselines in Natural Language Inference. *arXiv preprint arXiv:1805.01042*.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Association for Computational Linguistics (ACL)*.

M. Richardson, C. J. Burges, and E. Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 193–203.

Marta Sabou, Kalina Bontcheva, Leon Derczynski, and Arno Scharl. 2014. Corpus annotation through crowdsourcing: Towards best practice guidelines. In *LREC*, pages 859–866.

R. Schwartz, M. Sap, Y. Konstas, L. Zilles, Y. Choi, and N. A. Smith. 2017. The effect of different writing tasks on linguistic style: A case study of the ROC story cloze task. In *Computational Natural Language Learning (CoNLL)*.

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*.

Alane Suhr, Stephanie Zhou, Iris Zhang, Huajun Bai, and Yoav Artzi. 2018. A corpus for reasoning about natural language grounded in photographs. *CoRR, abs/1811.00491*.

A. Talmor, J. Herzig, N. Lourie, and J. Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *North American Association for Computational Linguistics (NAACL)*.

A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman. 2017. NewsQA: A machine comprehension dataset. In *Workshop on Representation Learning for NLP*.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Omar F. Zaidan and Chris Callison-Burch. 2011. Crowdsourcing translation: Professional quality from non-professionals. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.