Joint End-to-End Semantic Proto-role Labeling

Elizabeth Spaulding[2*] and **Gary Kazantsev**[1] and **Mark Dredze**[1,3]
Bloomberg, L.P., New York, NY USA[1]
University of Colorado Boulder, Boulder, CO, USA[2]
Computer Science, Johns Hopkins University, Baltimore, MD USA[3]
elizabeth.spaulding@colorado.edu, mdredze@cs.jhu.edu, gkazantsev@bloomberg.net

Abstract

Semantic proto-role labeling (SPRL) assigns properties to arguments based on a series of binary labels. While multiple studies have evaluated various approaches to SPRL, it has only been studied in-depth as a standalone task using gold predicate/argument pairs. How do SPRL systems perform as part of an information extraction pipeline? We model SPRL jointly with predicate-argument extraction using a deep transformer model. We find that proto-role labeling is surprisingly robust in this setting, with only a small decrease when using predicted arguments. We include a detailed analysis of each component of the joint system, and an error analysis to understand correlations in errors between system stages. Finally, we study the effects of annotation errors on SPRL.

1 Introduction

Semantic analyses of text have been framed (Gildea and Jurafsky, 2000) as extracting structured information in the form of predicates, arguments, and their relations, often called semantic roles. Multiple schemas have been proposed for structuring semantic roles, each with its own benefits and challenges. Semantic proto-roles (Dowty, 1991) offer a way to decompose traditional inventories of thematic roles into simple properties that are both easier to annotate and more generalizable to unseen arguments. These emerge from Dowty's proto-role theory, which assigns properties to arguments based on how agent-like (*volition*, *sentience*) or patient-like (*change of state*, *was used*) they are. For example, in the sentence "The boy threw a rock," categorical role inventories assign argument "boy" the role Agent, and argument "rock" the role Patient. Work on decompositional semantics[1] has formulated the task of semantic proto-role labeling as the assignment of 14 different binary properties to arguments (Reisinger et al., 2015a).

Multiple systems have been proposed for automatically assigning proto-roles to predicate-arguments pairs in text (Opitz and Frank, 2019; Rudinger et al., 2018; Teichert et al., 2017; Tenney et al., 2019), which have established the feasibility and best practices for semantic proto-role labeling (SPRL). At the same time, this task continues to be either treated in total isolation, assuming gold predicates and arguments, or included in Universal Dependency Semantics (UDS) parsing pipelines (Stengel-Eskin et al., 2020, 2021), which has not included fine-grained analysis on semantic proto-role properties themselves. How well does SPRL work when integrated into a semantic extraction pipeline? Are earlier errors compounded by SPRL? Are the same tokens challenging for each stage of the pipeline?

We answer these questions by constructing a joint multi-task model for identifying predicates and arguments, and assigning proto-role properties. Competitive with state-of-the-art for *both* dependency- and span-based SPRL evaluation, a careful component-wise analysis of our system allows us to make the following contributions. 1) Despite SPRL labeling errorful predicate argument predictions, our results are still competitive with having gold predicates and arguments, and far surpass the only previous work that predicts proto-roles jointly with predicates and arguments. Future work should include SPRL scores with predicted arguments. 2) Errors in predicates and arguments do not negatively affect SPRL because the same tokens that are challenging for argument identification are challenging for SPRL. 3) We find that most SPRL errors come from arguments with annotator disagreement, which suggests that these are inherently hard; removing unskilled annotators doesn't change performance, suggesting that conflict alone is not the source of the problem. We discuss implications for future work on SPRL after our analysis.

[*] Work done during an internship at Bloomberg.

[1] http://decomp.io/

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 723–736
July 9-14, 2023 ©2023 Association for Computational Linguistics

2 Semantic Proto-Roles

Semantic role labeling (SRL) was first formulated as a natural language understanding task by Gildea and Jurafsky (2000) and quickly proliferated (Surdeanu et al., 2003; Xue and Palmer, 2004; Pradhan et al., 2005) as resources and common evaluation frameworks were introduced (Carreras and Màrquez, 2004, 2005; Pradhan et al., 2007; Surdeanu et al., 2008; Hajič et al., 2009). SRL assigns relationships or roles of an argument and its predicate. Various labels build on different linguistic theories: label inventories that are small and coarse versus large and fine-grained. Common roles include Agent, Patient, Goal, and Location. Neural-based systems that assign SRL jointly with other related tasks, such as predicate and argument identification, perform just as well or better than models trained on only SRL (Conia et al., 2021; Blloshmi et al., 2021; He et al., 2018; Strubell et al., 2018; Li et al., 2019).

Argument identification can be formulated as dependency-based (find the argument's syntactic head) or span-based (find the entire argument span). The CoNLL 2004 and 2005 shared tasks (Carreras and Màrquez, 2004, 2005) used spans: an argument is correct only if *all* argument tokens are correctly identified with the correct argument role. CoNLL 2008 and 2009 (Surdeanu et al., 2008; Hajič et al., 2009) used a dependency-based method, which only requires that the syntactic head of the argument be tagged with the correct argument role. Understandably, span-based is more challenging and scores lag behind dependency-based systems (Li et al., 2019).

SPRL (Dowty, 1991) offers an alternative (Reisinger et al., 2015a) by decomposing traditional semantic roles into properties. The two proto-roles are "cluster-concepts" called Proto-Agent and Proto-Patient, which each correspond to an inventory of properties. Certain properties (such as *volition* or *instigation*) tend to belong to Proto-Agents, while others (such as *change of state* and *was used*) tend to belong to Proto-Patients. This analysis offers increased granularity but without sparsification of the training data.

The state-of-the-art SPRL dependency-based system fine-tunes BERT (Devlin et al., 2019) with a multi-layer Perceptron to assign labels using a linear combination of different BERT layer embeddings (Tenney et al., 2019). For span-based, the leading system uses an attention-based ensemble and trainable "argument marker embeddings" to indicate which tokens are arguments (Opitz and Frank, 2019). Stengel-Eskin et al. (2020) and Stengel-Eskin et al. (2021) jointly predict UDS graph structures (i.e., the spans of predicates and arguments) with all UDS properties, including semantic proto-roles. Both use a sequence-to-graph transductive model, and Stengel-Eskin et al. (2021) is able to improve the transductive model by integrating transformer architecture. Systems are rarely evaluated in both dependency- and span-based settings, and none have been evaluated on anything but gold predicates and arguments.

3 Data

We report results on two English-language datasets for SPRL: SPR1 (Reisinger et al., 2015b) and SPR2 (White et al., 2016). SPR1 contains 4,912 Wall Street Journal sentences from PropBank (Kingsbury and Palmer, 2002; Palmer et al., 2005; Gildea and Palmer, 2002) annotated by a single annotator based on a set of 18 proto-role properties. 9,738 arguments were annotated for the likelihood (on a Likert scale from 1 to 5) that a property holds for that argument. SPR2 contains 2,758 English Web Treebank (Bies et al., 2012) sentences annotated for a smaller set of 14 properties using a revised, streamlined protocol. In this release, multiple annotators ensured two-way redundancy for each property judgment.

Following previous work (Opitz and Frank, 2019; Rudinger et al., 2018; Teichert et al., 2017; Tenney et al., 2019), we formulate SPRL as a 18 (SPR1) or 14 (SPR2) way multi-label binary classification problem and map Likert labels $\{1, 2, 3\}$ to 0, and $\{4, 5\}$ to 1. The task has also been formulated as a regression problem in which SPRL scores are predicted as continuous values (Opitz and Frank, 2019; Rudinger et al., 2018), but we do not include this formulation as a part of our analysis. We additionally map judgments labeled "inapplicable" to 0 to ensure consistency with previous work. We use standard train/dev/tests splits provided in the data. We additionally do analysis on inter-annotator agreement in SPR2 shown in Appendix B.2.

4 Joint End-to-End SPRL

We construct a joint end-to-end SPRL system based on BERT (Devlin et al., 2019) with classification heads for each sub-task (Figure 1). We fine-tune

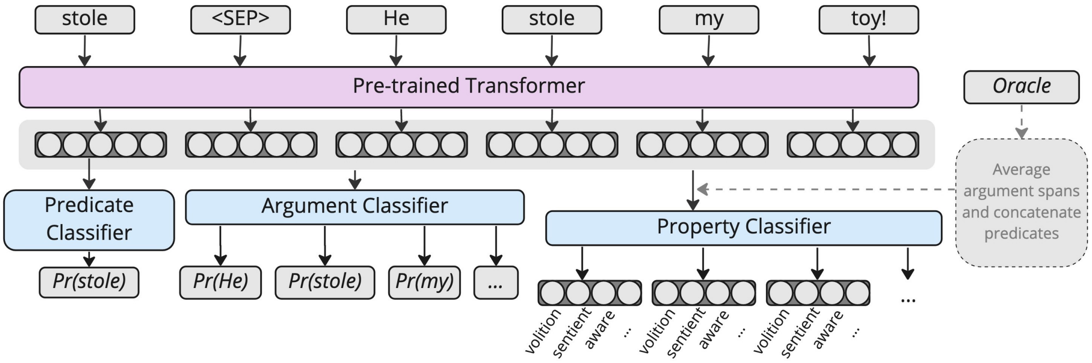

Figure 1: Our model fine-tunes a pre-trained transformer. Only the embedding for the first token of each sequence is passed through the predicate classifier. Input to the argument and property classifiers is each token embedding concatenated with the embedding for the predicate.

BERT and the encoder parameters are shared across tasks. We favor BERT as opposed to newer encoders for a direct comparison to Tenney et al. (2019) and Opitz and Frank (2019).[2]

We construct representations of the input sentences to enable the system to efficiently identify predicates, arguments, and SPRL. For each sentence, we construct an instance with a candidate predicate prepended onto the sentence with a separator. We use linear classification heads[3] with sigmoid functions to produce classification probabilities for each token. We place a classification head on the prepended predicate token to determine if it is a predicate. For argument and SPRL, we use separate classification heads on each token in the sentence. Dependency-based models predict argument for the dependency head only, whereas span-based models use an IOE tagging and softmax outputs. The resulting argument-tagged subwords are pooled and concatenated with the predicate representation taken from the sentence into the SPRL binary classification heads. Since we create an input for each possible predicate, we reduce the number of examples originating from incorrect predicates by only using predicate candidates that were verbs. Full model and training details appear in Appendix A.3.

Our model architecture is very similar to Tenney et al. (2019), except we extend it for predicate and argument identification. Additionally, we do not use a linear combination of BERT layers, instead taking BERT's last layer of BERT.[4]

	SPR1		SPR2	
	macro	micro	macro	micro
Dependency Prediction				
This Paper	72.7	85.5	65.0	83.3
This Paper + Arg prediction	73.7	85.5	**68.1**	82.4
This Paper + Predicate + Arg prediction	**74.0**	85.3	64.7	**83.8**
Rudinger et al. (2018)	71.1	83.3	-	-
Tenney et al. (2019)	-	**86.1**	-	**83.8**[5]
Span Prediction				
This Paper	71.4	83.7	65.0	82.9
This Paper + Arg prediction	71.8	84.1	65.7	81.9
This Paper + Predicate + Arg prediction	73.0	**84.3**	65.2	81.6
Opitz and Frank (2019)	69.3	82.0	69.7	83.4
Opitz and Frank (2019) + BERT	**73.8**	83.5	67.5	**83.9**
Stengel-Eskin et al. (2020) Transductive parser	-	-	65.4	-
Stengel-Eskin et al. (2021)[6] TFMR + EN + BERT	-	-	**69.8**	83.3
Span Prediction (Ensembles)				
Opitz and Frank (2019) Ensemble	72.1	83.6	**70.9**	84
Opitz and Frank (2019) Ensemble + BERT	**77.5**	**86.8**	69.9	**84.9**

Table 1: F1 scores on proto-role property prediction when using gold predicates and arguments. Our models were trained concurrently with argument classification, and also predicate classification tasks (where indicated). All other models besides Stengel-Eskin et al. (2020, 2021) do not perform joint inference. **Bold** indicates best performance within a section.

5 Experiments

We run multiple experiments to isolate the behavior of different components of our system, such as training on only SPRL, as well as the full pipeline. For all experiments, we train both a dependency-based and span-based model.

[2] Newer encoders would likely do better, but our goal is an analysis and not obtaining new state-of-the-art scores.

[3] We experimented with multi-layer Perceptrons (ReLU activation) as heads, but found no consistent improvement.

[4] In 2019, it was common to use a combination of layer embeddings based on ELMO, but since then systems use only the final output layer. Results in Table 1 verify that layer combinations are not necessary for this task.

[5] This score is for BERT-base, which we use in our experiments. The authors gained a small boost of 0.3 F1 points by using BERT-large.

[6] Scores for the system reported by Stengel-Eskin et al. (2021) were obtained using their released code. These scores represent the best-performing setting: TFMR + EN, tuning the top 8 layers of BERT.

	SPR1			SPR2		
	Recall		***Strict F1***	***Recall***		***Strict F1***
Dependency Prediction	*Preds*	*Arg Heads*	*Properties*	*Preds*	*Arg Heads*	*Properties*
Gold Predicates	-	93.2	77.5 *(-8)*	-	95.7	79.0 *(-3.4)*
+ Predicate Prediction	94.8	95.2	78.0 *(-7.3)*	83.4	97.8	81.8 *(-2)*
	Recall		***Strict F1***	***Recall***		***Strict F1***
Span Prediction	*Preds*	*Arg Spans*	*Properties*	*Preds*	*Arg Spans*	*Properties*
Gold Predicates	-	91.6	78.8 *(-5.3)*	-	86.7	77.8 *(-4.1)*
+ Predicate Prediction	95.2	91.2	77.6 *(-6.7)*	92.4	87.6	74.7 *(-6.9)*

Table 2: F1 scores for each prediction task using *strict* scoring. Since it is more important to see how many predicates and arguments the model guessed correctly, we report on recall for predicates and arguments. For the proto-role properties, the micro-F1 score is reported. *Red italics* show the drop in F1 score from gold scores.

Scoring SPRL is typically reported as micro/macro averaged F1 across the individual SPR binary properties. We report Gold F1 scores that assume the previous stages of the pipeline produced correct predicates and arguments. However, when considering SPRL run on predicted predicates and arguments, we need to adjust the scoring such that we penalize the SPRL score due to mistakes earlier in the pipeline. For other tasks, such as entity linking, we can simply mark a link as "missed" if we fail to recognize an entity with a NER system. However, because SPRL is a binary classification task, scoring is more complex.

We consider two different SPRL scoring methods for false negative predicate or arguments: (1) a lenient score that assumes 0 for all properties, which means that missed arguments do not have any of the properties. (2) A strict score that forces the label *incorrect* for all properties, which assumes we get all property predictions wrong thereby marking them incorrect. We do not modify the SPRL score for false positive arguments (for which there are no gold labels) since this would change the set of arguments over which each run of the system is evaluated. For example, in the sentence "Bob sat on the chair and I laid on the ground", if the model predicts that "ground" is an argument for "sat", then the model would produce property predictions for "ground" even though there are no annotations for this token. Those predictions are ignored entirely, because it is not guaranteed that other runs will also include predictions for this token.

We evaluate each component of our system separately to determine the effects of the pipeline. (1) Train only the SPRL classifiers using gold predicates and arguments. (2) Train arguments and SPRL assuming gold predicates. (3) Train predicates, arguments, and SPRL using inputs first filtered to consider only verbs as predicates. We replicate this training for both span and dependency-based predictions. For each setting, we evaluate under different conditions by decoding assuming gold or predicted labels from earlier in the pipeline.

6 Results

We present an overview of the results for our joint system, but full results appear in Appendix B. Table 1 shows our SPRL system compares favorably to previous work. We show three systems: trained on only SPRL, trained on arguments and SPRL, and trained on predicates, arguments, and SPRL. In all cases, we decode SPRL predictions assuming gold predicates and arguments. Our model matches or surpasses previous span and dependency results on SPR1, but lags slightly behind on span-based SPR2. This confirms previous work that found SPR2 more difficult than SPR1, perhaps because SPR2 has less data and more complex predicates and arguments.

Table 2 shows performance using our strict pipeline scoring, in which we map proto-role property predictions to *incorrect* for false negative arguments. The drop in F1 from using gold labels is shown in red. While we do worse in a pipeline, with the largest gap being 8 points for dependency-based SPR1, jointly learning predicates slightly improves strict F1 performance on SPRL in the dependency-based models, but degrades performance in the span-based models. Furthermore, SPR1 suffers a larger drop in the strict scoring regime than SPR2, perhaps because SPR2 models were already predicting many of the "harder" arguments incorrectly.

How are SPRL errors related to mistakes earlier in the pipeline? SPRL performance was much lower for arguments that would have been missed earlier in the pipeline. (See Table 3.) Table 2 shows this effect: models with smaller drops from gold were *already* making errors on incorrect arguments, whereas models with larger drops were likely bet-

		Correct args		Incorrect args	
Dataset	**Model**	*Size*	*F1*	*Size*	*F1*
SPR1	Dependency	17,712	86.4	1,296	67.3
	+ Predicate prediction	18,090	86.0	918	65.0
	Span	17,406	84.9	1,602	71.9
	+ Predicate prediction	17,334	85.3	1,674	69.0
SPR2	Dependency	7,770	82.9	350	68.4
	+ Predicate prediction	7,938	84.0	182	74.0
	Span	7,042	83.1	1,078	70.5
	+ Predicate prediction	7,112	82.8	1,008	70.9

Table 3: F1 score (micro-averaged) on subsets of arguments the model predicted correctly and arguments the model predicted incorrectly. All models in this table concurrently predict arguments and SPRL. Models that additionally predict predicates are noted.

ter at handling “difficult” examples. These difficult examples seem to correlate with annotation difficulty. We measure the performance of the system on annotation subsets based on the difference in Likert scores from the annotator. The larger the disagreement in Likert scores betweeen annotators, the worse the model performance (Appendix B.1.2.) To rule out the role of poor annotators, we removed those who had low inter-annotator agreement with others. However, this had almost no effect on F1, suggesting that it is the examples themselves that are challenging, and not the quality of the annotations. Perhaps these arguments are challenging for both tasks, or possibly the BERT encoder learns a poor representation of them. Fortunately, this means that when arguments are correctly discovered, SPRL does a good job on them and that correcting errors may improve both tasks.

Additionally, since we follow previous work by collapsing proto-role annotations marked “inapplicable” into the 0 class, we investigate the effect of excluding “inapplicable” property annotations in Table 7 and find a consistent boost of at least 3 F1 points by excluding inapplicable annotations, suggesting future work may benefit from handling applicability judgements differently, such as in Stengel-Eskin et al. (2020, 2021), who use a hurdle model in which a first classifier determines whether or not a property applies before making the property value judgement. Together, the effects of Likert disagreements and inapplicability of proto-role annotations additionally suggests that normalizing the different annotator responses, as in White et al. (2020), who use a mixed effects model, might lead to better outcomes in SPRL. See Appendix B.1 for a more detailed analysis of all results.

7 Discussion

Our end-to-end SPRL system demonstrates the efficacy of SPRL when combined with a full system. We are competitive with both span-based and dependency-based models and find that joint identification of predicates and arguments still produces a high-performing SPRL system. Future work should evaluate this setting, using both span- and dependency-based models and our proposed scoring method. Furthermore, our work points to the need for focused improvement on challenging arguments, which is harming both argument identification and SPRL. Do these errors show the limits of SPRL since annotators also get them wrong? Do we need better encoder training? Will downstream tasks that consume SPRL labels be robust to these errors? What is the feasibility of a reinforcement learning system that trains on the model’s own output? These questions remain for future work.

Limitations

Our analysis of the behavior of SPRL focused on intrinsic task scores. Higher SPRL scores suggest a better system. In practice, we do not yet understand how these scores affect downstream uses of SPRL labels. Furthermore, SPRL datasets are relatively small and are English only. As we are limited to the labels in the existing datasets, we are uncertain about how our results would generalize to larger datasets, new domains, and other languages.

Ethics Statement

When deploying a system such as ours on real text, e.g., news, one should carefully consider the implications of labeling real entities with certain proto-role properties. For example, answering the question of whether or not an actor *instigated* some action could have serious ramifications in the real world. Care should be taken so that such cases might be, for example, flagged for human review.

Acknowledgements

We thank Elias Stengel-Eskin, Benjamin Van Durme, Igor Malioutov, and Leslie Barrett for their helpful comments and feedback throughout conversations surrounding the project. We additionally acknowledge anonymous Bloomberg employees for assistance in reviewing the paper. Finally, we thank the ACL reviewers for their careful consideration and invaluable feedback.

References

Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English web treebank.

Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. 2021. Generating senses and roles: An end-to-end model for dependency- and span-based semantic role labeling. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3786–3793. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 89–97, Boston, Massachusetts, USA. Association for Computational Linguistics.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

Simone Conia, Andrea Bacciu, and Roberto Navigli. 2021. Unifying cross-lingual semantic role labeling with heterogeneous linguistic resources. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 338–351, Online. Association for Computational Linguistics.

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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

David Dowty. 1991. Thematic Proto-Roles and Argument Selection. *Language*, 67(3):547–619. Publisher: Linguistic Society of America.

Daniel Gildea and Daniel Jurafsky. 2000. Automatic labeling of semantic roles. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 512–520, Hong Kong. Association for Computational Linguistics.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 239–246, USA. Association for Computational Linguistics.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado. Association for Computational Linguistics.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369, Melbourne, Australia. Association for Computational Linguistics.

Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02)*, Las Palmas, Canary Islands - Spain. European Language Resources Association (ELRA).

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Juri Opitz and Anette Frank. 2019. An argument-marker model for syntax-agnostic proto-role labeling. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 224–234, Minneapolis, Minnesota. Association for Computational Linguistics.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Comput. Linguist.*, 31(1):71–106.

Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Semantic role chunking combining complementary syntactic views. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 217–220, Ann Arbor, Michigan. Association for Computational Linguistics.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 87–92, Prague, Czech Republic. Association for Computational Linguistics.

Drew Reisinger, Rachel Rudinger, Francis Ferraro, Craig Harman, Kyle Rawlins, and Benjamin Van Durme. 2015a. Semantic Proto-Roles. *Transactions of the Association for Computational Linguistics*, 3:475–488.

Drew Reisinger, Rachel Rudinger, Francis Ferraro, Craig Harman, Kyle Rawlins, and Benjamin Van Durme. 2015b. Semantic proto-roles. *Transactions of the Association for Computational Linguistics*, 3:475–488.

Rachel Rudinger, Adam Teichert, Ryan Culkin, Sheng Zhang, and Benjamin Van Durme. 2018. Neural-Davidsonian semantic proto-role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 944–955, Brussels, Belgium. Association for Computational Linguistics.

Elias Stengel-Eskin, Kenton Murray, Sheng Zhang, Aaron Steven White, and Benjamin Van Durme. 2021. Joint universal syntactic and semantic parsing. *Transactions of the Association for Computational Linguistics*, 9:756–773.

Elias Stengel-Eskin, Aaron Steven White, Sheng Zhang, and Benjamin Van Durme. 2020. Universal decompositional semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8427–8439, Online. Association for Computational Linguistics.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 8–15, Sapporo, Japan. Association for Computational Linguistics.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177, Manchester, England. Coling 2008 Organizing Committee.

Adam Teichert, Adam Poliak, Benjamin Van Durme, and Matthew Gormley. 2017. Semantic proto-role labeling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2016. Universal decompositional semantics on Universal Dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723, Austin, Texas. Association for Computational Linguistics.

Aaron Steven White, Elias Stengel-Eskin, Siddharth Vashishtha, Venkata Subrahmanyan Govindarajan, Dee Ann Reisinger, Tim Vieira, Keisuke Sakaguchi, Sheng Zhang, Francis Ferraro, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2020. The universal decompositional semantics dataset and decomp toolkit. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5698–5707, Marseille, France. European Language Resources Association.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 88–94, Barcelona, Spain. Association for Computational Linguistics.

	SPR1	SPR2
Train		
Precision	45.1	49.4
Recall	100	100
F1	62.2	66.2
Dev		
Precision	47.4	53.1
Recall	100	100
F1	64.3	69.4
Test		
Precision	45.1	54.2
Recall	100	100
F1	62.1	70.3

Table 4: Results for the predicate filtering step for every split.

A Training details

We simplify the complex task of joint predicate-argument-proto-role learning, as the space of possible predicates, arguments, and proto-role labels is $O(|R|n^3)$ for a sentence of n tokens and a set of proto-role properties R. (There are $O(n)$ possible predicates and $O(n^2)$ possible argument spans.) First, we made the decision not to train the model on its own output—ie, we use an oracle to identify gold predicate and argument tokens so that non-predicate sequences and non-argument tokens are ignored in the loss step.

A.1 Pre-processing

We shift some of the complexity to the data processing step before any learning occurs by crafting sequences such that only one predicate is considered at a time: for example, the sentence "He stole my toy!" would be split into four separate data points:

```
He <SEP> He stole my toy!
stole <SEP> He stole my toy!
my <SEP> He stole my toy!
toy! <SEP> He stole my toy!
```

The model learns to focus on the first token as the candidate predicate of the sentence. For example, in the sequence `stole <SEP> He stole my toy`, the model must answer the questions: If *stole* is the predicate, what are the arguments of the sentence, and what are their proto-role properties? We truncate sequences to a fixed maximum length of 50 and pad shorter sequences to the right.

A.2 Predicate filtering

The number of training instances would be quite large if every token of every sentence was used as a predicate candidate, as above. So we undergo a predicate filtering step in which we only select tokens that are labeled as verbs (ie, anything with a POS tag beginning with `VB`) in the datasets. For every dataset and split, this initial predicate filtering step has a recall of 100. After filtering, the number of training instances in SPR1 is reduced to 8,999 from 83,789, and in SPR2, is reduced to 7,452 from 46,138. Model output for argument identification and SPRL on false positive predicates is ignored in the loss function and evaluations. Table 4 shows full results for the predicate filtering.

A.3 Hyperparameters

Using the hyperparameters from previous work as a starting point, we fine-tuned the learning rate and batch size and then kept them fixed based on the highest validation macro-F1 for final experiments. We report scores from a single run from each final experiment. We use a batch size of 8, run for 30 epochs with no early stopping, and choose scores based on the best validation macro-F1. Our learning rate is 0.00001. For each property, we apply loss weights equal to the inverse frequency of that property. Our model, which uses BERT-base, contains 109M trainable parameters, and took roughly 2-6 hours to train on a single GPU depending on the size of the dataset, whether or not we were predicting predicates, and whether or not we were predicting argument spans. We used the Pytorch Lightning[7] framework to build and train our model.

B Full results

The full proto-role property identification results for all model configurations using a linear classification head can be found arranged in a grid in Table 5. The grid shows the three training methods and the three scoring methods. For training, we have three columns indicating whether or not the model was trained to predict predicates, arguments, and proto-role properties. For training settings in which we do not train the model to predict predicates, note that we do not create sequences with incorrect predicates (ie, the model would never see the sequence `He <SEP> He stole my toy!`) and the model only sees instances with correct predicates.

[7] `https://www.pytorchlightning.ai`, Apache-2.0

						Dependency				Spans			
Train			Test			SPR1		SPR2		SPR1		SPR2	
P	*A*	*R*	*P*	*A*	*R*	*macro*	*micro*	*macro*	*micro*	*macro*	*micro*	*macro*	*micro*
✗	✗	✓	✗	✗	✓	72.7	85.5	65.0	83.3	71.4	83.7	65.0	82.9
✗	✓	✓	✗	✗	✓	73.7	85.5	68.1	82.4	71.8	84.1	65.7	81.9
✗	✓	✓	✗	✓	✓^	72.9	84.4	67.6	81.3	64.0	85.0	60.0	80.3
✗	✓	✓	✗	✓	✓*	62.2	77.5	64.1	79.0	59.4	78.8	57.1	74.8
✓	✓	✓	✗	✗	✓	74.0	85.3	64.7	83.8	73.0	84.3	65.2	81.6
✓	✓	✓	✗	✓	✓^	73.3	84.5	64.3	83.2	64.8	85.3	59.1	79.9
✓	✓	✓	✗	✓	✓*	64.7	79.5	62.1	81.9	60.0	78.8	56.5	74.8
✓	✓	✓	✓	✓	✓^	72.9	83.9	64.3	83.2	64.2	84.9	58.9	79.8
✓	✓	✓	✓	✓	✓*	63.1	78.0	62.0	81.8	59.0	77.6	56.4	74.7

Table 5: Full grid of results using a linear classification head. P/A/R refers to the three tasks: predicate/argument/role identification. In the column labeled "Train", we see which of the three tasks the model is trained to predict. In the column labeled "Test", we see which of the three tasks is accounted for in the score. Gold scores are shown in the top row of each section. ^ indicates lenient F1. * indicates strict F1.

			Dependency								Spans							
			SPR1				SPR2				SPR1				SPR2			
Train			*Preds*	*Arg Heads*			*Preds*	*Arg Heads*			*Preds*	*Arg Spans*			*Preds*	*Arg Spans*		
P	*A*	*R*	*F1*	*P*	*R*	*F1*	*F1*	*P*	*R*	*F1*	*F1*	*P*	*R*	*F1*	*F1*	*P*	*R*	*F1*
✗	✓	✓	-	77.1	93.2	84.4	-	82.1	95.7	88.4	-	62.3	91.6	74.1	-	70.8	86.7	78
✓	✓	✓	94.8	73.5	95.2	83	83.4	61.9	97.8	75.8	95.2	65.7	91.2	76.4	92.4	74.1	87.6	80.3

Table 6: Predicate and argument identification results for all joint models. We show all three of precision, recall, and F1 for argument identification. For dependency-based models, we show results for correctly retrieving only the argument head. For span-based, we only count the argument span as correctly retrieved if every single token in the span was correct.

For scoring, we show three different scoring methods: (1) gold scores, which assume correct predicates and arguments earlier in the pipeline, for direct comparison to previous work; (2) lenient scores, which assume 0 for all proto-role properties, treating SPRL as a "proto-role retrieval" task; and (2) strict scores, which map proto-role properties to the wrong label if predicates and arguments are falsely predicted as 0 earlier in the pipeline. We do not modify the SPRL score for false positives in predicate and argument identification since this would chnage the set of arguments over which the system is evaluated. The corresponding results for predicate and argument identification can be found in Table 6.

B.1 Evaluating on subsets of data

To attempt to tease out the reasons for various errors in the model predictions, we take varying subsets of the data and evaluate separately on each subset. We report sizes of the different subsets we evaluate in Table 8.

B.1.1 Arguments predicted correctly and incorrectly

To further investigate the question of how errors earlier in the pipeline propagate later in the pipeline, we take a subset of arguments which the model predicted correctly and a subset of arguments which the model predicted incorrectly, and calculate the F1 scores for each subset. We report these scores in Table 3. We see large differences in the F1 scores between these subsets, suggesting that arguments that are difficult for the model to identify are also difficult for proto-role property classification.

An example of an argument that all configurations of our SPR2 models struggled with is italicized in the sentence below, with the predicate in bold:

> I **like** *I Move CA - Los Angeles Movers*, they moved me before, but this time they were awesome :)

None of the models were able to retrieve this argument correctly (neither the head, nor the span). They all made mistakes on at least some proto-role

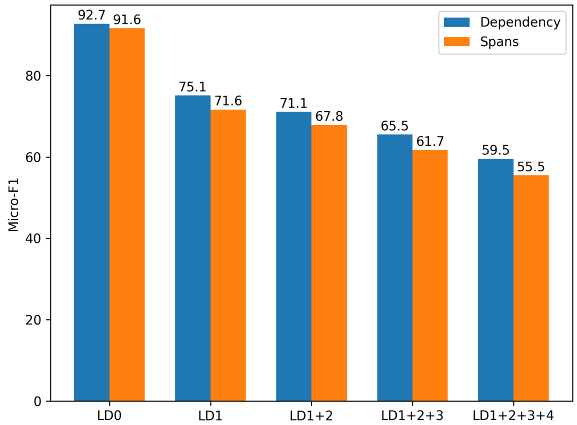

Figure 2: F1 score (micro-averaged) for the full joint SPR2 models on subsets of arguments based on the difference of Likert ratings given by two annotators.

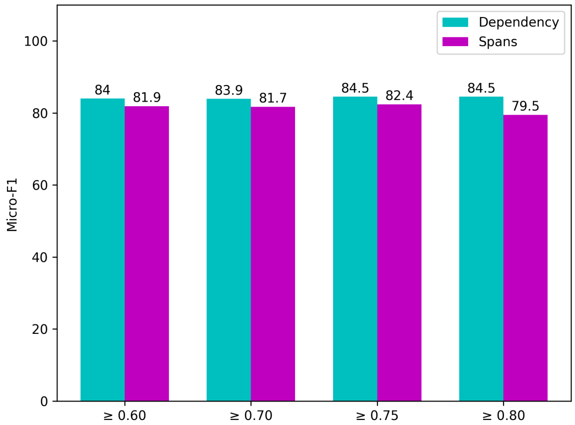

Figure 3: F1 scores (micro-averaged) for the full joint SPR2 models based on subsets of arguments requiring *both* annotators to have a minimum pairwise inter-annotator agreement. The x-axis shows the IAA requirements that determine the subsets.

property predictions: common mistakes among all configurations of the model included false negatives for *sentient*, false positives for *awareness*, and false positives for *change of location*.

Interestingly, only the span-based models predicted that the argument was not *sentient*, showing that the non-head tokens in the span confused the model. On the other hand, both the dependency-based models understandably predicted that the head of the argument, *Movers*, changed location, while the span-based models did not make this mistake.

We notice that this sentence might have been difficult for annotators to judge, so we proceed with evaluations of subsets based on annotator judgements and agreement to tease out the association between examples that are difficult for annotators and examples that are difficult for the model.

B.1.2 Differences in Likert ratings

For SPR2, which is doubly-annotated, we hypothesized that we could locate examples that are difficult for the model to classify by the difference between the two annotators' Likert ratings in each property judgement. We construct several different subsets of data, which we refer to as LDi. LDi is the subset of property annotations in which the difference between Likert ratings between two annotators is exactly i. We show F1 scores for different combinations of these subsets in Figure 2, and provide the sizes of each subset in Table 8. We see that the score on the subset containing only property judgements with complete agreement between annotators is far higher than all other scores. As we add property judgements with larger and larger disagreement between annotators, the scores drop substantially.

B.1.3 Pairwise inter-annotator agreement

In section B.2, we show inter-annotator agreement κ scores averaged over each property. We also calculate κ pairwise for each annotator and report these scores in Table 10. We then investigate the extent to which including annotations by annotators with low pairwise agreement affects F1 by creating subsets of data which excludes annotations by annotators with an inter-annotator agreement below some cutoff. We report these scores in Figure 3. Surprisingly, we note that excluding annotators with low pairwise inter-annotator agreement has almost no affect on the F1 score, suggesting that annotator “skill” is less important than the difficulty of each example in SPRL F1.

B.1.4 Applicability judgements

Finally, we investigate the extent to which the applicability judgements correlate with difficulty of property prediction. For both SPR1 and SPR2, an “applicable” judgement, indicating whether or not the proto-role property was applicable to the argument in the context of the sentence, was collected for each property in addition to the Likert judgements. As a reminder, annotations marked inapplicable were collapsed into the 0 class regardless of the Likert rating. Thus, in Table 7, we show F1 scores on the subset of annotations marked “applicable” by both annotators (or, in the case of SPR1, the single annotator) versus F1 scores on the entire

Dataset	**Model**	*Applicable*	*All*
SPR1	Dependency	88.9	85.5
	+ Predicate prediction	88.5	85.3
	Span	88.2	84.1
	+ Predicate prediction	88.2	84.3
SPR2	Dependency	86.3	82.4
	+ Predicate prediction	86.9	83.8
	Span	86.6	81.9
	+ Predicate prediction	86.0	81.6

Table 7: F1 score (micro-averaged) on the subset of property annotations marked "applicable" versus the F1 for all property annotations. In the case of SPR2, the subset only contains annotations where both annotators agreed the property was applicable.

Subset	**SPR1**	**SPR2**
A0	8,925	997
A1	10,083	1,775
A2	n/a	5,348
LD0	n/a	4,611
LD1	n/a	1,563
LD2	n/a	830
LD3	n/a	570
LD4	n/a	546

Table 8: Sizes of each subset used in evaluations for analysis. *Ai* is the subset of property annotations where exactly i annotators marked "Applicable" as "True." *LDi* is the subset of property annotations where the difference between Likert ratings between two annotators is exactly i.

dataset. We see a consistent boost of at least 3 F1 points by only evaluating on applicable annotations.

B.2 Inter-annotator agreement

A possible limitation of the currently available SPR data is relatively low average inter-annotator agreement. White et al. (2016) report an agreement of 0.617 using Spearman's rank correlation coefficient for SPR2. However, this agreement was measured over the Likert scores, which our model will not be predicting. We re-measured both the Likert data and the collapsed binary data using Cohen's kappa on a per-property basis. We see in Table 9 that when measuring agreement using Cohen's kappa, collapsing the Likert labels to $\{0, 1\}$ improves the agreement significantly, resulting in every property having at least $\kappa \geq 0.64$.

Dataset	**Property**	**Likert** κ	**Binary** κ
1&2	instigation	0.61	0.69
1&2	volition	0.77	0.86
1&2	awareness	0.82	0.88
1&2	sentient	0.82	0.88
1&2	change of location	0.59	0.71
1	exists as physical	-	-
1&2	existed before	0.74	0.79
1&2	existed during	0.79	0.86
1&2	existed after	0.68	0.76
1	created	-	-
1	destroyed	-	-
1&2	change of possession	0.66	0.80
1&2	change of state	0.59	0.66
1	stationary	-	-
1	location of event	-	-
1	physical contact	-	-
1&2	was used	0.59	0.66
1	pred changed arg	-	-
2	was for benefit	0.61	0.70
2	partitive	0.58	0.64
2	change of state continuous	0.65	0.67
	Average	**0.68**	**0.75**

Table 9: Proto-role properties and their inter-annotator agreement, measured using Cohen's kappa, where applicable. Note that because the SPR1 release was annotated with one annotator, the agreement scores only apply to SPR2.

We also calculated each annotator's Cohen's kappa score pairwise against every other annotator (and averaged). We then experimented with scoring our models on a subset of the data in which only judgements by annotators with a certain inter-annotator agreement were kept. The inter-annotator agreement scores used in these experiments can be found in Table 10.

Annotator ID	**Likert κ**	**Binary κ**	**# Annotations**
0	**0.43**	**0.42**	14
1	0.72	0.81	5,418
2	0.56	0.74	28
3	0.70	0.76	1,932
7	0.62	0.73	9,184
8	0.67	0.78	14
10	0.75	0.81	1,078
11	0.61	0.78	42
13	0.69	0.73	14
15	0.71	0.80	7,224
16	0.68	0.73	2,814
20	0.64	0.76	3,640
25	0.70	0.75	3,248
26	0.70	0.78	19,250
29	0.70	0.81	7,504
30	0.65	0.78	1,652
32	0.65	0.74	8,708
35	0.68	0.74	1,204
37	0.60	0.68	1,092
40	**0.81**	**0.85**	14
43	0.67	0.75	14,854
45	0.67	0.77	126
46	0.65	0.71	1,498
48	0.69	0.78	1,358
50	0.63	0.72	140
51	0.70	0.80	308
56	0.71	0.79	6,496
62	0.68	0.71	3,850
64	0.68	0.76	518
65	0.71	0.81	1,512
66	0.69	0.76	4,186
68	0.72	0.80	588
69	0.75	0.79	14
70	0.65	0.76	4,746
71	0.70	0.77	4,144
73	0.67	0.67	2,744
74	0.51	0.50	546
75	0.66	0.80	4,942
76	0.67	0.77	4,228
78	0.70	0.79	896
81	0.68	0.75	854
87	0.69	0.77	23,002
92	0.69	0.75	6,580
93	0.67	0.76	4,746
94	0.73	0.82	3,682
Average	**0.67**	**0.75**	

Table 10: Pairwise inter-annotator agreement measured with Cohen's kappa. *Italics* show the lowest κ value. **Bold** shows the highest κ value.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations (unnumbered, page 5)

☑ A2. Did you discuss any potential risks of your work?
Ethics (unnumbered, page 5)

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Section 1 - Introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4 and Appendix A

☑ B1. Did you cite the creators of artifacts you used?
Section 4 (BERT) and Appendix A.3 (Pytorch Lightning)

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Appendix A.3

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 3 - Data

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 3 - Data

C ☑ Did you run computational experiments?

Section 5 - Experiments, Appendix A

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix A.3

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix A.3

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Appendix A.3 - since we only report from a single run, we do not provide error bars around results

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix A.3

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.