Inducing Semantic Roles Without Syntax

Julian Michael[1] **and Luke Zettlemoyer**[1,2]

[1]Paul G. Allen School of Computer Science & Engineering, University of Washington
[2]Facebook AI Research
{julianjm,lsz}@cs.washington.edu

Abstract

Semantic roles are a key component of linguistic predicate-argument structure, but developing ontologies of these roles requires significant expertise and manual effort. Methods exist for automatically inducing semantic roles using syntactic representations, but syntax can also be difficult to define, annotate, and predict. We show it is possible to automatically induce semantic roles from QA-SRL, a scalable and ontology-free semantic annotation scheme that uses question-answer pairs to represent predicate-argument structure. By associating arguments with distributions over QA-SRL questions and clustering them in a mixture model, our method outperforms all previous models as well as a new state-of-the-art baseline over gold syntax. We show that our method works because QA-SRL acts as *surrogate syntax*, capturing non-overt arguments and syntactic alternations, which are central motivators for the use of semantic role labeling systems.[1]

Labels	Questions	
A1 (98%)	What is given?	.30
	What does something give something?	.21
	What does something give?	.20
	What is something given?	.11
A0 (98%)	What gives something?	.44
	What gives something something?	.27
	What gives something to something?	.08
A2 (94%)	What is given something?	.28
	What does something give something to?	.18
	What does something give something?	.14
	What is given?	.09
	What is something given to?	.07
TMP (46%), ADV (22%), MNR (12%)	When does something give something?	.20
	How does something give something?	.09
	When is something given?	.09
	When is something given something?	.09
PNC (30%), ADV (22%), TMP (14%)	Why does something give something?	.18
	Why does something give up something?	.07
	Why is something given something?	.07

Table 1: Roles for *give* produced by our final model. Core arguments are captured almost perfectly, exhibiting both passive and dative alternations.

1 Introduction

Semantic role labeling (SRL) requires extracting propositional predicate-argument structure from language, i.e., *who* is doing *what* to *whom*. Applications of SRL include information extraction (Christensen et al., 2011), machine reading (Wang et al., 2015), and model analysis (Tenney et al., 2019; Kuznetsov and Gurevych, 2020), and semantic roles form the backbone of many more general meaning representations (Banarescu et al., 2013; Abend and Rappoport, 2013).

The primary challenge, and promise, for SRL systems is to distill syntactically variable surface structures into semantic predicate-argument structures from an ontology (Palmer et al., 2005; Baker et al., 1998). However, ontologies and their associated training data require time and expertise to annotate and do not readily generalize to new domains, limiting their broad-coverage applicability. Prior work towards mitigating this problem includes unsupervised induction of semantic roles from syntactic representations (Lang and Lapata, 2010). However, the need for formal syntactic supervision retains some of the annotation and generalization difficulties of supervised SRL, and it has proven difficult to do much better than a simple syntactic baseline (Lang and Lapata, 2011). An alternative is to use an ontology-free annotation scheme like QA-SRL (He et al., 2015), which represents roles with natural language questions. While QA-SRL can be annotated at large scale (FitzGerald et al., 2018), many different QA-SRL questions may correspond to the same role, making it more

[1]Code, models, and a web interface to explore the results are available at https://github.com/julianmichael/qasrl-roles.

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4427–4442
August 1–6, 2021.

*The plane was **diverting** around weather formations over the Java Sea when contact with air traffic control (ATC) in Jakarta was **lost**.*

wh	aux	subj	verb	obj	prep	obj2	?	Answer
What	was		being diverted		around		?	*weather formations*
What	was		diverting				?	*The plane*
What	was		being diverted				?	*The plane*
What	was		lost				?	*contact with air traffic control*
Where	was	something	lost				?	*over the Java Sea*

Table 2: Example QA-SRL question-answer pairs from the development set of the QA-SRL Bank 2.0 (FitzGerald et al., 2018). Questions may be represented in a verb-agnostic way by recording the form of the verb in the **verb** slot (e.g., *stem*, *past participle*). Note that the syntax used in questions may differ from the syntax in the source sentence, for example in the above questions using *diverted* in its passive form.

difficult to use in downstream tasks.

We show how to overcome this difficulty, by automatically inducing an ontology of semantic roles corresponding to clusters of QA-SRL questions (see Table 1 for an example clustering). We use a model to predict a distribution over QA-SRL questions associated with each argument in a corpus, and cluster them to maximize likelihood under a simple model we call a *Hard Unigram Mixture*. Our model can be effectively optimized both by EM and greedy methods, which affords the benefits of tunable hierarchical clustering without sacrificing scalability (Section 3).

Experiments in semantic role induction (Section 4) show that our method outperforms all previous methods in the literature, as well as a new state-of-the-art baseline over gold syntax. This is despite requiring no formal syntactic supervision or theory, where the formalism used by previous work is highly informative of gold standard semantic roles (Section 5). We also present a detailed analysis (Section 6) showing why our method works: QA-SRL acts as *surrogate syntax*, removing (role-irrelevant) syntactic variation in the source text such as that from *non-overt* arguments (e.g., phrases extracted from relative clauses), while itself exhibiting (role-relevant) syntactic alternations which capture the behavior of verbal predicates (Table 1). Taken together, these results paint a path towards on-the-fly, data-driven construction of useful, interpretable ontologies of semantic structure.

2 Task Setting

The input to our task is a set of natural language sentences, where a subset of the tokens are marked as *predicates*. Each predicate has a set of *arguments*, and each argument x corresponds to a set of spans $x = \{s_1, \ldots, s_m\}$ in the predicate's sentence.[2]

An ontology of semantic roles is a set of *frames* (corresponding to semantic predicates), and each frame has a set of associated *roles* (corresponding to participants in the event or state denoted by its frame). There may also be a set of *modifier roles* (e.g., location or time) which can appear with any frame. In supervised semantic role labeling, each predicate in the input data must be assigned to one of the frames in a given ontology, and each of a predicate's arguments must be assigned roles from its frame (or modifier roles). In semantic role induction, our task is to produce both the ontology and these assignments.

We follow prior work (Lang and Lapata, 2010) in treating semantic role induction as a clustering problem and assuming a single frame per predicate lemma.[3] Given input data marked with predicates and their arguments, we cluster the arguments for each predicate into sets corresponding to semantic roles. We may then compare these clusters to gold labels using clustering metrics (Section 4.3).

[2]Previous work (Lang and Lapata, 2010) assumes a syntactic dependency tree and marks each argument by its syntactic head, which allows for features based on argument lemmas and dependency paths. We instead assume sets of argument spans, but no syntax tree; this allows for features based on spans (such as QA-SRL questions). Both approaches are ways of featurizing the same gold arguments.

[3]Some ontologies, like FrameNet (Baker et al., 1998), define frames that span multiple lemmas (e.g., *buy* and *sell* share a *Commercial Transaction* frame), whereas others like PropBank (Palmer et al., 2005) use frames which are specific to each lemma, denoting something closer to word sense. In our case, assuming a single frame per lemma simplifies modeling and allows us to compare to previous work. However, modeling predicate sense is an important problem for future work, as we will suggest in Section 6.3.

3 Modeling

Our model treats each argument x as a set of counts of QA-SRL questions,[4] denoted $\phi(x)$. We produce these counts from a trained QA-SRL question generator (Section 3.1) and cluster them by maximizing their likelihood under a mixture model (Section 3.2) using a hybrid of flat and hierachical clustering (Section 3.3).

3.1 Generating QA-SRL Features

For each argument x of a predicate, we leverage a trained QA-SRL parser to generate pseudocounts $\phi(x)$ of simplified QA-SRL questions, which will form the input features for the clustering step.

Simplified QA-SRL Example QA-SRL questions are shown in Table 2. These questions contain information which is not directly relevant to semantic roles, such as tense, aspect, modality, and negation. Since this creates sparsity for our model, we remove it as a preprocessing step. In particular, we replace the **aux** and **verb** slot values with either *is* and *past participle* (for passive voice), _ and *present* (for active voice when **subj** is blank), or *does* and *stem* (for active voice when **subj** is present). We also replace all occurrences of *who* and *someone* with *what* or *something*.

Generating Question Counts Let p denote a predicate, s denote a span, and q denote a simplified QA-SRL question. To generate our question count vectors ϕ, we reproduce the QA-SRL question generator of FitzGerald et al. (2018), which generates a distribution $\mathrm{P}(q \mid p, s)$ over QA-SRL questions conditioned on a predicate p and answer span s in a sentence. This model uses a BiLSTM encoder, concatenating the output representations of span endpoints and feeding them into a custom LSTM decoder which models the QA-SRL slot values in sequence. We modify the model to use BERT (Devlin et al., 2019) features as input embeddings for the BiLSTM (details in Appendix A).

Recall from Section 2 that an argument x consists of a set of spans from its sentence. We generate question counts $\phi(x) \in \mathbb{R}^{|q|}_{\geq 0}$ by taking the mean

$$\phi(x) = \frac{1}{|x|} \sum_{s \in x} \mathrm{P}(q \mid p, s),$$

where $\mathbb{R}_{\geq 0}$ denotes the nonnegative real numbers and $|q|$ is the number of possible simplified QA-SRL questions. Since $|q|$ is large, to make this tractable we approximate $\mathrm{P}(q \mid p, s)$ with beam search, using a sparse representation and assigning counts of 0 to questions outside the beam.

3.2 Objective

Let $\mathbf{X} = \{x_1, \ldots, x_n\}$ be the set of input arguments for clustering. Our goal is a clustering $\mathbf{C} = \{C_1, \ldots, C_k\}$ which is a partition of $\mathbf{X}$. We model each argument's questions $\phi(x)$ as being drawn from a mixture model over latent roles, each corresponding to a cluster $C \in \mathbf{C}$. We maximize likelihood under this model, which we call a Hard Unigram Mixture, with the addition of a connectivity penalty which encourages roles not to appear twice for the same predicate instance.

The Hard Unigram Mixture (HUM) Recall that $\phi : \mathbf{X} \to \mathbb{R}^d_{\geq 0}$ assigns question pseudocounts to each $x \in \mathbf{X}$. Let π denote a probability distribution over $\{1, \ldots, k\}$ and θ a distribution over $\{1, \ldots, d\}$. We propose the *Hard Unigram Mixture* loss

$$\mathcal{L}^{\text{HUM}}_{\lambda}(\mathbf{C}) = -\log \mathrm{P}(\mathbf{X} \mid \mathbf{C}) - \lambda \log \mathrm{P}(\mathbf{C}),$$

where

$$\mathrm{P}(\mathbf{X} \mid \mathbf{C}) = \prod_i^k \max_\theta \prod_{x \in C_i} \mathrm{P}(\phi(x) \mid \theta)$$

is the *data likelihood* and

$$\mathrm{P}(\mathbf{C}) = \max_\pi \prod_i^k \pi_i^{||C_i||}$$

is the *clustering likelihood*, writing $||C||$ for the sum of the ϕ counts in a cluster C. The data likelihood prefers more, smaller clusters, the clustering likelihood prefers fewer clusters, and λ is a hyperparameter that trades off between them.[5]

Connectivity Penalty Let $p(x)$ denote the predicate instance corresponding to an argument x. We propose a *connectivity penalty*

$$\mathcal{L}^{\text{cp}}(\mathbf{C}) = \frac{1}{2} \sum_i^k \sum_{x_1, x_2 \in C_i} \delta(p(x_1) = p(x_2)),$$

[4] As of now, this model only works for English, as QA-SRL is only defined and annotated in English.

[5] Here, $\mathcal{L}^{\text{HUM}}_1$ is equivalent to the negative log likelihood under the maximum likelihood estimate of a mixture of unigrams model (Nigam et al., 2000) constrained to hard assignments $\mathbf{C}$; hence the name *Hard Unigram Mixture*. Further theoretical and empirical comparison to prior work is provided in Appendix G.

where δ is the indicator function, which discourages clusterings where multiple arguments of the same predicate instance are assigned the same role. This assumption has also been leveraged by prior models (Lang and Lapata, 2011; Titov and Klementiev, 2012).

Loss Function Our full loss is then

$$\mathcal{L}_\lambda(\mathbf{C}) = \mathcal{L}_\lambda^{\text{HUM}}(\mathbf{C}) + \mathcal{L}^{\text{cp}}(\mathbf{C})$$

with the single hyperparameter λ.

3.3 Hybrid Clustering

We optimize $\mathcal{L}_\lambda$ in three steps: flat pre-clustering, greedy merging, and tuned splitting. This approach provides us with both the efficiency benefits of flat clustering and the relative determinism, interpretability and tunability of hierarchical clustering.

Flat Pre-Clustering For pre-clustering, we minimize $\mathcal{L}_0$ via hard EM. To avoid likelihoods of 0 in $\mathcal{L}_0^{\text{HUM}}$, we smooth our estimates of θ using a Dirichlet prior. To optimize $\mathcal{L}^{\text{cp}}$ via EM, we draw x_1 from the previous iteration's clustering in order to compute the contribution of each x_2 to the loss. With sufficiently large k, this can produce a high-precision clustering in $O(nk)$ time to serve as input to the merging step.

Greedy Merging After pre-clustering, we produce a binary cluster tree by iteratively merging pairs of clusters which greedily minimize $\mathcal{L}_0$. Since $\lambda = 0$, the loss grows monotonically when merging clusters. The loss at each merge can be efficiently updated by maintaining maximum likelihood estimates θ for each cluster.

Tuned Splitting Finally, we iteratively split the cluster tree produced by the merging stage. At each step, we split the cluster C_i with the lowest log data likelihood per item $\frac{\log \mathrm{P}(C_i|\mathbf{C})}{|C_i|}$. We then choose the clustering which minimizes $\mathcal{L}_\lambda$, with $\lambda > 0$ tuned during model development.[6]

4 Experimental Setup

Data We run experiments on the distribution of PropBank (Palmer et al., 2005) provided for the CoNLL 2008 Shared Task (Surdeanu et al., 2008). We use the same setup as previous work, removing arguments annotated with reference (R-) and continuation (C-) roles, keeping only verbal predicates,[7] and using the development set for model development and the training set for testing.

Our one preprocessing difference from previous work is that instead of using the dependency-based SRL annotations provided in the CoNLL 2008 dataset, we use full answer spans, which we reconstruct by aligning the CoNLL 2008 data back to the original annotations in the Penn Treebank (Marcus et al., 1993) and PropBank.[8]

4.1 Models

***HUM of QA-SRL Questions* (HUM-QQ)** We train a QA-SRL parser on the expanded set of the QA-SRL Bank 2.0 (FitzGerald et al., 2018) using the architecture described in Section 3.1. In the pre-clustering step, we estimate $k = 100$ clusters. For tuned splitting, we choose λ to maximize performance on the development set. Hyperparameters are detailed in Appendix B.

SYNTF This model assigns each argument to a cluster corresponding to the label of its syntactic dependency to its parent, using the syntactic formalism provided in CoNLL 2008 Shared Task data. Past work has found SYNTF to be a strong baseline (Lang and Lapata, 2011).

Prior Work We compare to Bayesian generative modeling (Titov and Klementiev, 2012, BAYES), which is state-of-the-art on gold syntax, and an embedding-based method (Luan et al., 2016, SYMDEP/ASYMDEP) which is state-of-the-art using automatic syntax. These as well as all other prior approaches (e.g., Lang and Lapata, 2011; Titov and Khoddam, 2015; Woodsend and Lapata, 2015) crucially rely on syntactic features.

4.2 Auxiliary Clustering Rules

For SYNTF and HUM-QQ, we experiment with several auxiliary clustering rules.

Lexical Rules We employ three lexical rules, each producing a separate cluster for all arguments whose spans exactly match a phrase contained in the rule's lexicon. Our rules are for negation (5

[6] A comparison of this method against a constant-k baseline and oracle upper bound is given in Appendix E.

[7] While we ignore nominal predicates, our method naturally generalizes to nominalizations, which are provided with QA-SRL annotations in QANom (Klein et al., 2020).

[8] Using gold spans is necessary in order to compare to previous work and use the CoNLL 2008 dataset for evaluation of role induction. In a more realistic setting where gold argument spans are not available, we could use the span detector of FitzGerald et al. (2018) to construct argument spans.

phrases), modals (23 phrases), and discourse modifiers (55 phrases). These lexica were written to correspond to the AM-NEG, AM-MOD, and AM-DIS roles on the basis of the PropBank annotation guidelines (Babko-Malaya, 2005) and development set.[9]

Passive to Active Conversion We also propose a syntactic rule that applies only to SYNTF, where we the transform the dependencies as follows:

- The LGS label, meaning "logical subject," is a dependency label given for *by*-phrases modifying a passive verb whose object denotes what is normally the subject of the verb's active form (Surdeanu et al., 2008). We change this to SBJ.
- Passive voice can be detected when the predicate verb is in past participle form (part-of-speech tag VBN) and its syntactic parent is a *be*-verb (part of speech VC, lemma "be"). In these cases, we change the syntactic label of any SBJ dependents into OBJ.

4.3 Metrics

Purity/Collocation To compare with previous work, we follow Lang and Lapata (2010) in using purity and collocation based F1 score for our main evaluation. Purity measures cluster homogeneity: it assigns to each cluster the gold label for which it has the most points, and then measures the proportion of points which have their cluster's assigned label. Collocation measures cluster concentration: it assigns each gold label to the cluster which contains the most of its points, and then measures the proportion of points which are in their gold label's assigned cluster. These are calculated independently for each verb and averaged, weighing each verb by its number of argument instances. The harmonic mean of the final results is reported as an F1 score.

B^3 For deeper analysis, we use the B^3 (*B-cubed*) family of clustering metrics (Bagga and Baldwin, 1998). B^3 precision and recall are the precision and recall of each point's predicted cluster against its gold cluster, averaging over points. In comparison to purity and collocation, these metrics are tougher and more discriminative between clusterings, respecting important constraints like the cluster completeness constraint of Rosenberg and Hirschberg (2007), among others (Amigó et al., 2009). B^3 also allows us to reliably report scores along slices of the data for analysis purposes, as well as account for each slice's contribution to the total error. We report full B^3 results for our models in Appendix F and encourage future work to use these as the primary metrics.

[9] Full lexica for these rules are provided in Appendix C.

Model	PU	CO	F1	ΔF1
Gold Syntax				
SYNTF	81.6	77.8	79.6	0.0
+ lex	85.2	79.8	82.4	+2.8
+ pass→act	83.6	80.8	82.2	+2.6
+ all rules	87.3	**83.1**	**85.2**	**+5.6**
BAYES (SotA)	**88.7**	78.1	83.0	+3.4
ASYMDEP	85.6	78.3	81.8	+2.2
Automatic Syntax				
BAYES	86.2	72.7	78.8	-0.8
SYMDEP (SotA)	81.9	**76.6**	**79.2**	-0.4
Automatic QA-SRL				
HUM-QQ	80.9	83.4	82.1	+2.5
− conn. penalty	79.0	82.7	80.8	+1.2
+ lex	**85.4**	**88.8**	**87.1**	**+7.5**

Table 3: Main results. The addition of a few simple rules to the SYNTF baseline puts it significantly above existing approaches, and incorporating these rules into our QA-SRL-based model pushes performance even further, despite not using gold syntax at all. Evaluation numbers for baselines besides SYNTF are drawn directly from prior work.

5 Results

Main results are shown in Table 3. Our auxiliary rules put SYNTF significantly above the state of the art for gold syntax (with 85.2 F1 versus 83.0). HUM-QQ surpasses it with 87.1 F1 in the best case, despite not using gold syntax at all.

5.1 A Stronger Syntactic Baseline

For SYNTF, the addition of either lexical (negation, modal, and discourse) rules or the passive-to-active conversion produce competitive models, covering over 75% of the gap from baseline to BAYES. Used together, our rules bring the score to 85.2 F1, surpassing BAYES by 2.2 points. Table 5 breaks down these improvements by measuring B^3 performance on relevant roles.

For the lexical rules, we find that the negation and modal rules nearly completely capture their

B^3 F1	A0	A1	A2	A3	A4	Args	TMP	ADV	MNR	LOC	PNC	CAU	Mods	All
SYNTF + lex	78	71	63	**55**	**67**	73	**87**	**51**	**60**	**81**	**65**	**67**	**74**	74
HUM-QQ + lex	**90**	**87**	**69**	54	65	**85**	78	39	50	55	56	36	61	**82**
% Err \ Freq	.26	.37	.09	.01	.01	.74	.07	.04	.03	.03	.01	.01	.18	1.0
SYNTF + lex	.23	.41	.13	.03	.02	.79	.04	.06	.04	.02	.01	.01	.18	1.0
HUM-QQ + lex	.15	.26	.14	.04	.02	.61	.08	.12	.07	.06	.02	.02	.38	1.0

Table 4: B^3 F1 scores on the training set for the most common labels, excluding NEG, MOD, and DIS.

roles, with the discourse rule providing significant improvements as well. In contrast, previous models have struggled with these roles, as reported by Lang and Lapata (2011, Table 4, NEG and DIS roles). However, this is better seen as a shortcoming of the evaluation than the models: these roles are relatively uninteresting from the perspective of semantic role induction, as they are closed-class, not specific to particular predicates, and don't correspond to a semantic argument or modifier of the event denoted by the predicate. It might have been reasonable to exclude these arguments from the task at the outset, but instead, using our rules can mostly account for them while maintaining some comparability to prior work.

The passive-to-active conversion also produces a sizable gain, particularly on the core roles A0 and A1 (Table 5). Titov and Klementiev (2012) informally note that the BAYES model learns some syntactic alternations; of these, the passive alternation is perhaps the most impactful as it can apply to any transitive verb. What we've found is that a simple rule accounting for the passive construction in the syntax provided to the BAYES model can account for a large majority of its gains.

These results provide extra context in which to interpret the existing literature on semantic role induction. The fact that our simple auxiliary rules bring the syntactic baseline beyond the existing state of the art raises questions about whether the performance differences between previously published models are due to their relative abilities in capturing their intended phenomena — such as selectional restrictions and distributions over argument heads (Lang and Lapata, 2014) — or capturing these rules. It is not clear how much of the 5.2 F1 gain over SYNTF from our auxiliary rules is redundant with previous models. It seems likely that applying our rules to them would produce a result competitive with HUM-QQ, but it would still rely on gold syntax. Our focus is the utility of QA-SRL as features; indeed, it is also conceivable that applying a hierarchical model like BAYES to QA-SRL features would bring further improvements as well.

	B^3 F1 Score				
Model	NEG	MOD	DIS	A0	A1
SYNTF	41	45	50	78	71
+ all rules	**98**	**98**	**80**	**83**	**78**
Frequency	.01	.04	.03	.26	.37

Table 5: Breakdown of B^3 F1 scores on the training set for the labels most relevant to our auxiliary rules. The lexical rules capture AM-NEG, AM-MOD, and AM-DIS very well, and the active/passive rule significantly improves performance on A0 and A1, which are by far the most frequent role labels in the data. A rule-by-rule performance breakdown is provided in Appendix D.

5.2 Superiority Without Syntax

HUM-QQ benefits disproportionately from the lexical rules, with a 5 F1 gain as opposed to the 2.8 F1 gain for SYNTF. This is because PropBank's NEG, MOD, and DIS arguments almost never occur in QA-SRL, so they get nonsense questions from the model (see Appendix J, Table 12).[10] However, even the baseline model with no lexical rules or connectivity penalty surpasses the performance of the baselines using automatic syntax, all of which fall short of SYNTF on gold.[11] With these additions, HUM-QQ sets a new state of the art beyond our enhanced SYNTF baseline, with 87.1 F1.

Table 4 compares our model to SYNTF + lex on the most common roles using B^3. HUM-QQ greatly improves over SYNTF on core arguments

[10] In practice, when using arguments predicted by a QA-SRL span detector (FitzGerald et al., 2018), we can remove the lexical rules entirely since the corresponding arguments will not be present.

[11] To be fair, these models use the automatic parses provided with the CoNLL 2008 data, which were produced by MaltParser (Nivre et al., 2006) at the time. Using state-of-the-art methods to predict the parses today would almost certainly improve the semantic role induction results, but probably not past gold parses.

(73→85 F1), but performs worse on modifiers (74→61). Since core arguments make up 74% of arguments in the corpus, HUM-QQ brings a large improvement overall (74→82) and core arguments still account for a majority of its error (at 61%).

SYNTF's high performance on modifiers can be traced back to representational choices in the CoNLL 2008 Shared Task syntax (Surdeanu et al., 2008), which uses several dependency types that are semantic in nature, such as TMP, LOC, MNR, and DIR, among others. These often correlate well with gold modifier role labels, especially TMP (87 F1) and LOC (81 F1).[12] This fact has led some prior work, e.g., Titov and Klementiev (2012), to use these dependency labels as clusters directly, so as to avoid the need to model modifier roles and instead focus on core arguments. Since we eschew syntactic features, we are forced to recover PropBank modifier roles from the ground up, making the task more difficult (explored more in Section 6.2).

6 What does QA-SRL Encode About Semantic Roles?

Semantic roles are traditionally characterized as abstractions over syntactic arguments and modifiers (Gruber, 1965; Fillmore, 1968). Despite their deep entanglement with syntax, we have found that significant improvements in semantic role induction are possible without explicit syntactic analysis of the sentence, instead leveraging distributions of QA-SRL questions for each argument. In this section, we show that this is because QA-SRL questions provide *surrogate syntax*, recapitulating the aspects of syntax that are important for semantic roles (Section 6.1). Where QA-SRL questions fail to capture aspects of PropBank semantic roles, this arises in part from ontological differences with PropBank on modifiers (Section 6.2) and limitations of our experimental setup ignoring predicate sense (Section 6.3).

6.1 Surrogate Syntax

HUM-QQ brings the largest improvement over SYNTF on core arguments A0 and A1. To investigate this, we identify the verbs which saw the greatest increase in B^3 F1 score on each role individually. What we find is that QA-SRL works by acting as *surrogate syntax*: it removes much of the (role-irrelevant) syntactic variation in the source text, while still exhibiting (role-relevant) syntactic alternations which capture the syntactic behavior of the predicate verb.

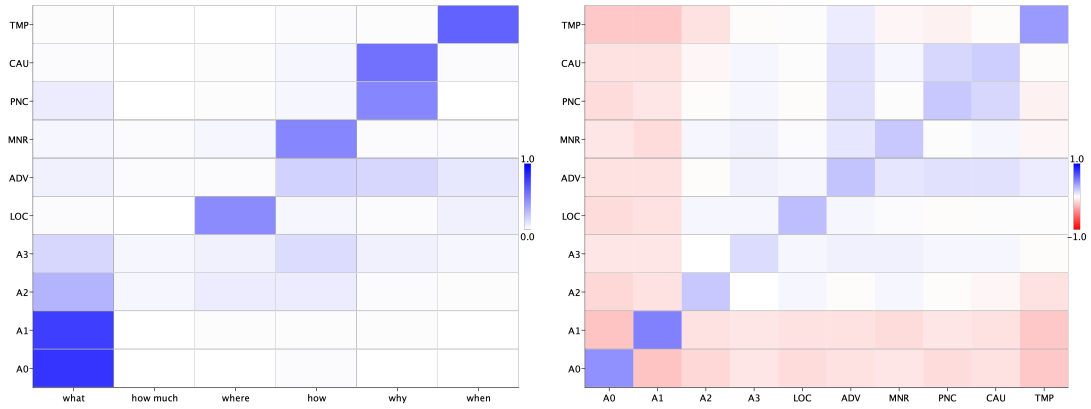
Reducing Syntactic Variation For A0, the three verbs with the greatest improvement from SYNTF to HUM-QQ are *compete*, *conduct*, and *connect*, all with gaps of over 40 F1.[13] For each of these, their A0 arguments have a wide range of syntactic functions assigned by SYNTF, with SBJ less than 50% of the time — despite the fact that where the A0 role is present, it is designed to correspond to the grammatical subject (Babko-Malaya, 2005). We found that this is because these verbs frequently have *non-overt* subjects, which are not direct syntactic dependents of the predicate in CoNLL 2008 syntax (74% of a random sample of 30 sentences with A0 arguments of these three verbs, 10 from each; see Appendix H.1). They appear in phrases like 'two competing *objectives*' (with adjectival clauses), 'urging *directors* to conduct a fair auction' (with control verbs), or '*a maze of halls* that connects film rooms' (with relative clauses). In these cases, the SYNTF baseline does poorly, as the correspondence between the SBJ dependency and A0 role only holds consistently for overt subjects.

In contrast, HUM-QQ assigns the vast majority of A0 arguments in these cases with questions that put the *wh*-word in subject position, e.g., *What competes with something?* or *What conducts?* Here, QA-SRL removes much of the syntactic variation from the source text and recovers something close to the underlying grammatical relation between the argument and the verb, while also providing information about the verb's subcategorization frames (e.g., the presence of an object in *What connects something?*), aiding in recovery of the semantic role.

Capturing Syntactic Alternations For A1, The verbs with the greatest improvement are *propose*, *prefer*, *price*, and *relate*, with a gap of >50 F1 between models. Of the top 50 such verbs, 48 are transitive with A1 as the transitive object (see Appendix H.2). In these cases, the passive alternation allows the argument to be asked about in either the subject (*What is proposed?*) or object (*What does something propose?*) position. We find that QA-SRL does this, frequently combining questions

[12] See Lang and Lapata (2014, Table 2) for a detailed contingency table.

[13] To reduce variance from low-frequency verbs, we measure this gap after smoothing their precision and recall with 10 counts of the weighted aggregate for the model.



(a) Distribution of *wh*-words for each role.

(b) Normalized PMIs between gold roles.

Figure 1: Cooccurrence between gold role labels and *wh*-words in QA-SRL (left) or each other in HUM-QQ's predicted clusters (right). The distributions of *wh*-words are normalized per role, and NPMI between gold labels is chance-corrected, where negative values (red) are clustered apart more often than by chance, and positive values (blue) are preferentially grouped together.

about passive subject and active object into one role: for 62% of the top 50 verbs, the cluster corresponding to A1 gives greater than 20% probability *each* to passive subject and active object questions. This happens because the Hard Unigram Mixture objective clusters together distributions whose uncertainty is spread over the same set of elements, which here correspond to syntactic alternations. As an example, Table 1 shows the induced clusters for *give*, which exhibit both passive and dative alternations; *give* gained 31 F1 on A1 in HUM-QQ.

6.2 Mismatched Modifiers

HUM-QQ struggles to identify PropBank modifier roles, and it has room for improvement on trailing arguments like A2 and A3. In QA-SRL, the semantics of these roles are primarily expressed by the initial *wh*-word, such as *when*, *where*, *why*, *how*, etc. Figure 1a shows the distribution of *wh*-words appearing for each role in the training set. To a large extent, each role is concentrated on a corresponding *wh*-word, but there are exceptions. A2, A3, and AM−ADV are widely spread between *wh*-words, and *how* and *why* account for a significant portion of questions for several roles each. See Appendix J, Table 11 for full questions.

To visualize how this affects clustering results, Figure 1b shows the normalized pointwise mutual information (NPMI; Bouma, 2009) between gold labels in HUM-QQ's predicted clusters (see Appendix I for how this is calculated). While A0 and A1 are distinguished well from all other roles, the trailing arguments A2 and A3 are not well distinguished from modifiers, reflecting the difficulty of the argument–adjunct distinction for these arguments, which often have similar meanings to modifiers and form a significant error case for supervised labelers (He et al., 2017). AM−ADV tends to be confused with other modifier roles, which reflects its definition in the PropBank guidelines as a sort of "catch-all" role for meanings not captured in the other modifiers (Babko-Malaya, 2005). Finally, AM−CAU (*cause*) and AM−PNC (*purpose, not cause*) tend to be confused with each other, since they both elicit *why* questions.

Argument–Adjunct Distinction Scores are significantly lower for trailing core arguments A2−4 than for A0 and A1. Since part of the problem seems to be confusion with modifier roles (Figure 1b), we conduct an oracle experiment to enforce the argument–adjunct distinction by doubling the size of the feature space to $\phi(x) \in \mathbb{R}^{2|q|}_{\geq 0}$ and projecting gold core arguments and modifiers into orthogonal subspaces.

Results are shown in Table 6 (+ gold arg/adj). The oracle boosts performance by 3 points, with particular focus on trailing arguments A2 (69→78) and A4 (65→78), as well as modifiers AM−ADV (39→47), AM−MNR (50→57), and AM−LOC (55→61). However, overall performance on modifiers is still far below the syntactic baseline. Given the coarse semantics of English *wh*-words in comparison to PropBank modifier roles (Figure 1a), it may be that finer-grained features are necessary to significantly increase performance on modifiers.

B^3 F1	A0	A1	A2	A3	A4	Args	TMP	ADV	MNR	LOC	PNC	CAU	Mods	All
SYNTF + lex	78	71	63	55	67	73	**87**	**51**	60	**81**	65	**67**	**74**	74
HUM-QQ + lex	90	87	69	54	65	85	78	39	50	55	56	36	61	82
+ gold arg/adj	91	89	78	65	78	88	77	47	57	61	58	35	64	85
+ gold sense	91	90	74	58	75	87	80	43	55	64	63	47	65	84
+ both	**92**	**92**	**83**	**70**	**81**	**90**	81	**51**	**64**	69	**66**	46	69	**87**

Table 6: Breakdown of B^3 F1 scores on the training set for the most common labels in our ablation studies. The first two rows are repeated from Table 4.

6.3 Scrambled Senses

Despite core arguments significantly improving under HUM-QQ, they remain the largest source of error. To investigate this, we examine the verbs with the worst F1 on core arguments. The top verbs are *go*, *settle*, *confuse*, *turn*, and *follow*, with <60 F1. Half of the top 20 have 4 or more predicate senses annotated in PropBank, where different senses often manifest their roles differently: for example, the subject is A0 when *settling with the IRS* (sense 2), but A1 when *settling into a new job* (sense 3). To quantify this, we run an oracle experiment where we induce roles for each verb sense separately instead of each verb lemma. Results are shown in Table 6 (+ gold sense). Performance improves particularly on trailing arguments A2, A3 and A4, which tend to differ greatly in meaning and realization for different predicate senses. A combined oracle (+ both) shows that the gains are mostly complementary with those from the argument/adjunct distinction oracle. These results suggest that future work on semantic role induction should prioritize modeling predicate senses.

7 Conclusion

We have shown that QA-SRL provides a way to do state-of-the-art semantic role induction without the need for formal syntax. It works by providing *surrogate* syntax: it captures long-distance dependencies to non-overt arguments and exhibits syntactic alternations which allow us to detect varied ways of expressing the same role. These results suggest that QA-SRL can provide some of the practical benefits of sophisticated syntactic formalisms that have separate layers of functional structure, like Combinatory Categorial Grammar (Steedman, 1996, 2000), Head-Driven Phrase Structure Grammar (Pollard and Sag, 1994), or Lexical Functional Grammar (Bresnan et al., 2015) — but without grammar engineering or expert data annotation.

One challenge is that QA-SRL is currently only defined for English. Future work may benefit from our lessons about the utility of surrogate syntax when designing similar annotation methodologies for other languages; combining this with insights from existing work on grammar development for diverse languages (Bender et al., 2002) may be key.

While formal ontologies of semantic roles and syntax are difficult to formulate and scale, our results show how it may be comparatively feasible to formulate, scale, and build robust models for the *phenomena* that such ontologies are meant to explain. QA-SRL exhibits enough of these phenomena that a relatively simple model over it (the Hard Unigram Mixture in Section 3) yields state-of-the-art induced semantic roles which are interpretable and linguistically meaningful. This suggests that identifying and gathering supervision for more phenomena (e.g., those related to word sense or modifier semantics) in a relatively theory-agnostic way, then building models grounded in linguistic theory, may be a promising avenue for future work. This general approach has recently been applied to syntax as well, for example leveraging constituency tests (Cao et al., 2020) and naturally-occurring bracketings (Shi et al., 2021).

The fact that discrete structures can be reliably derived from ontology-free annotation schemes like QA-SRL can potentially inform future efforts to construct large-scale ontologies of semantic structure. QA-SRL has the further benefit over traditional SRL of including a broader scope of *implicit* arguments than those addressed by supervised systems, as shown by Roit et al. (2020). Taken together, our results suggest that with the right kind of annotation scheme, it should be possible to construct rich semantic ontologies in new domains, without expert curation and in a data-driven, linguistically motivated way.

Acknowledgements

Thanks to the anonymous reviewers and Victor Zhong for their helpful comments.

References

Omri Abend and Ari Rappoport. 2013. Universal Conceptual Cognitive Annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–238, Sofia, Bulgaria. Association for Computational Linguistics.

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval*, 12(4):461–486.

Olga Babko-Malaya. 2005. Propbank annotation guidelines.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 79–85, Montreal, Quebec, Canada. Association for Computational Linguistics.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Emily M. Bender, Dan Flickinger, and Stephan Oepen. 2002. The grammar matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *COLING-02: Grammar Engineering and Evaluation*.

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. In *GSCL*.

Joan Bresnan, Ash Asudeh, Ida Toivonen, and Stephen Wechsler. 2015. *Lexical-functional syntax*. John Wiley & Sons.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Unsupervised parsing via constituency tests. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4798–4808, Online. Association for Computational Linguistics.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2011. An analysis of open information extraction based on semantic role labeling. In *K-CAP*.

Grzegorz Chrupała. 2012. Hierarchical clustering of word class distributions. In *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 100–104, Montréal, Canada. Association for Computational Linguistics.

Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *27th Annual Meeting of the Association for Computational Linguistics*, volume 16, pages 76–83. 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.

Charles J. Fillmore. 1968. The case for case. In Emmon Bach and Robert T. Harms, editors, *Universals in Linguistic Theory*. Holt, Rinehart & Winston.

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-scale QA-SRL parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2051–2060, Melbourne, Australia. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *NIPS*.

Jeffrey S. Gruber. 1965. *Studies in Lexical Relations*. Ph.D. thesis, Massachusetts Institute of Technology.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Ayal Klein, Jonathan Mamou, Valentina Pyatkin, Daniela Stepanov, Hangfeng He, Dan Roth, Luke Zettlemoyer, and Ido Dagan. 2020. QANom: Question-answer driven SRL for nominalizations. In *Proceedings of the 28th International Conference*

on Computational Linguistics, pages 3069–3083, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ilia Kuznetsov and Iryna Gurevych. 2020. A matter of framing: The impact of linguistic formalism on probing results. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 171–182, Online. Association for Computational Linguistics.

Joel Lang and Mirella Lapata. 2010. Unsupervised induction of semantic roles. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947, Los Angeles, California. Association for Computational Linguistics.

Joel Lang and Mirella Lapata. 2011. Unsupervised semantic role induction via split-merge clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1117–1126, Portland, Oregon, USA. Association for Computational Linguistics.

Joel Lang and Mirella Lapata. 2014. Similarity-driven semantic role induction via graph partitioning. *Computational Linguistics*, 40(3):633–669.

Kenton Lee, T. Kwiatkowski, Ankur P. Parikh, and Dipanjan Das. 2016. Learning recurrent span representations for extractive question answering. *ArXiv*, abs/1611.01436.

Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37:145–151.

Yi Luan, Yangfeng Ji, Hannaneh Hajishirzi, and Boyang Li. 2016. Multiplicative representations for unsupervised semantic role induction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 118–123, Berlin, Germany. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Julian Michael, Jan A. Botha, and Ian Tenney. 2020. Asking without telling: Exploring latent ontologies in contextual representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6792–6812, Online. Association for Computational Linguistics.

K. Nigam, A. McCallum, S. Thrun, and Tom Michael Mitchell. 2000. Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39:103–134.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fernando Pereira, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of English words. In *31st Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Columbus, Ohio, USA. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. 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 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Carl Pollard and Ivan A Sag. 1994. *Head-driven phrase structure grammar*. University of Chicago Press.

Paul Roit, Ayal Klein, Daniela Stepanov, Jonathan Mamou, Julian Michael, Gabriel Stanovsky, Luke Zettlemoyer, and Ido Dagan. 2020. Controlled crowdsourcing for high-quality QA-SRL annotation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7008–7013, Online. Association for Computational Linguistics.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic. Association for Computational Linguistics.

Tianze Shi, Ozan İrsoy, Igor Malioutov, and Lillian Lee. 2021. Learning syntax from naturally-occurring bracketings. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2941–2949, Online. Association for Computational Linguistics.

Noam Slonim and Naftali Tishby. 1999. Agglomerative information bottleneck. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, pages 617–623, Cambridge, MA, USA. MIT Press.

R. Srivastava, Klaus Greff, and J. Schmidhuber. 2015. Training very deep networks. In *NIPS*.

Mark Steedman. 1996. *Surface Structure and Interpretation*. MIT Press.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press.

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.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ivan Titov and Ehsan Khoddam. 2015. Unsupervised induction of semantic roles within a reconstruction-error minimization framework. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1–10, Denver, Colorado. Association for Computational Linguistics.

Ivan Titov and Alexandre Klementiev. 2012. A Bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 12–22, Avignon, France. Association for Computational Linguistics.

Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2015. Machine comprehension with syntax, frames, and semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 700–706, Beijing, China. Association for Computational Linguistics.

Kristian Woodsend and Mirella Lapata. 2015. Distributed representations for unsupervised semantic role labeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2482–2491, Lisbon, Portugal. Association for Computational Linguistics.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China. Association for Computational Linguistics.

A QA-SRL Question Generator

We reproduce FitzGerald et al. (2018)'s architecture, encoding sentences with a stacked alternating LSTM (Zhou and Xu, 2015) with highway connections (Srivastava et al., 2015) and recurrent dropout (Gal and Ghahramani, 2016), and representing spans by concatenating the output embeddings of their endpoints (Lee et al., 2016). The question generator is a specialized LSTM decoder which only outputs the tokens allowed in each QA-SRL slot. The current predicate is indicated by an embedded binary feature input to BiLSTM encoder, and answer span representations are input at each step of the LSTM decoder. We make two changes from FitzGerald et al. (2018): 1) As opposed to GloVe (Pennington et al., 2014) or ELMo (Peters et al., 2018), We embed the inputs with BERT-base (Devlin et al., 2019) in the 'feature' style with a learned scalar mix over layers, and 2) we additionally concatenate the output embedding of the predicate to the input of the LSTM decoder.

B Hyperparameters

QA-SRL Question Generator The BiLSTM encoder uses a hidden size of 300, 4 layers, 0.1 recurrent dropout probability, and a 100-dimensional predicate indicator embedding. The LSTM decoder has a 100-dimensional hidden state and predicts QA-SRL slots with 200-dimensional embeddings via an MLP with a 100-dimensional hidden layer. We train on all QA pairs in the QA-SRL Bank 2.0 expanded training set using BERT's variant of Adam (Kingma and Ba, 2015) with a learning rate of $5\mathrm{e}{-5}$ and batch size of 32, selecting the model with minimal perplexity on the expanded development set. To produce our feature vectors ϕ, we decode questions with a beam size of 20 and a minimum probability cutoff of 0.01.

Flat Pre-Clustering We perform flat clustering with 100 clusters, skipping this step for verbs with 100 arguments or less. We use a concentration parameter of $\alpha = 0.01$ (i.e., uniform base measure with a sum of 0.01) and do 5 random restarts, each running until the loss decreases by less than $1\mathrm{e}{-5}$ per iteration, and choose the run that yields the lowest loss.

Model	PU	CO	F1	ΔF1
SYNTF	81.6	77.8	79.6	0.0
+ negation	82.8	77.8	80.2	+0.6
+ modals	83.0	79.8	81.3	+1.7
+ discourse	82.6	77.8	80.1	+0.5
+ pass→act	83.6	80.8	82.2	+2.6
+ all rules	87.3	83.1	85.2	+5.6

Table 7: Detailed results for auxiliary rules on SYNTF.

Tuned Splitting Our final model (HUM-QQ + lex) uses $\lambda = 0.35$.

C Rule Lexica

Here we list the full lexica for the auxiliary clustering rules described in Section 4.2.

Negation 5 items: *n't*, *never*, *no*, *no longer*, *not*.

These are drawn directly from the PropBank guidelines (Babko-Malaya, 2005, p. 32).

Modals 23 items: *'d*, *'ll*, *'ve*, *able*, *ca*, *can*, *can't*, *could*, *going*, *gon*, *gonna*, *have*, *may*, *might*, *must*, *ought*, *shall*, *should*, *used*, *will*, *wo*, *won't*, *would*.

Note the inclusion of *have*, *used*, *able*, and *going*, which are parts of phrasal modals (e.g., *have to*), which are included in AM-MOD according to the PropBank guidelines (Babko-Malaya, 2005, p. 32).

Discourse 55 items: *after all*, *ah*, *also*, *and*, *and so*, *as a result*, *as we've seen before*, *as well*, *but*, *certainly*, *damn*, *either*, *for example*, *for instance*, *for one*, *for one thing*, *frankly*, *furthermore*, *gosh*, *hence*, *however*, *in addition*, *in any case*, *in any event*, *in contrast*, *in fact*, *in other words*, *in particular*, *in that case*, *in this case*, *in turn*, *indeed*, *instead*, *ironically*, *moreover*, *nonetheless*, *of course*, *oh gosh*, *oh my god*, *oh my gosh*, *on the other hand*, *or*, *particularly*, *rather*, *regardless*, *similarly*, *so*, *specifically*, *thereby*, *therefore*, *though*, *thus*, *too*, *uh*, *um*.

Note the inclusion of some interjections, (*ah*, *oh my gosh*, etc.), which are included in AM-DIS according to the PropBank guidelines (Babko-Malaya, 2005, p. 31).

D Auxiliary Rule Performance Breakdown

In Table 7, we provide a more detailed accounting of the improvements that arise from our auxiliary rules described in Section 4.2 and Table 5.

Tuning Method	PU	CO	F1
Constant $k = 6$	83.9	86.7	85.3
$\lambda = 0.35$	85.4	88.8	87.1
F1 Oracle	**87.6**	**89.6**	**88.6**

Table 8: Comparison of methods to determine the number of clusters for each verb. All reported numbers are for HUM-QQ+ lex.

Setting	Objective
$\lambda = 1$	Mixture of Unigrams Likelihood
$\lambda = 0$	Jensen-Shannon Divergence
$\lambda = -1$	Mutual Information

Table 9: Objectives reproduced by the HUM loss for different settings of λ, described in Appendix G.

The negation and discourse rules bring precision improvements, likely because they mostly have ADV dependencies outgoing. The modal rule improves both precision and recall because modals have many different kinds of outgoing dependencies, due to their status as heads of clauses (which can serve in many syntactic capacities). Finally, the passive alternation rule aids precision by splitting SBJ between active and passive uses, and aids recall by grouping LGS with the active SBJ and passive SBJ with active OBJ. This mainly affects the core argument labels A0 and A1, as shown in Table 5 — especially A1, as we also find for QA-SRL questions in Section 6.1.

E Tuned Splitting Evaluation

Our model has a single parameter λ which determines the number of clusters for each verb via the tradeoff between the data likelihood and clustering likelihood. We compare this to a constant baseline (the same number of clusters for all verbs) and an oracle upper bound which chooses the split that maximizes the purity/collocation F1 score for each verb independently. As shown in Table 8, we improve on the constant baseline by 1.8 points (85.3→87.1), but fall short of the oracle by 1.5 points (87.1→88.6). There is room for improvement, but errors in the tuning step may not be the most significant factor to concern future work.

F B^3 Results

Results using B^3 metrics on models we tested are shown in Table 10.

Model	B^3P	B^3R	F1	ΔF1
Gold Syntax				
SYNTF	74.7	68.3	71.3	0.0
+ lex	79.1	70.4	74.5	+3.2
+ pass→act	77.4	72.1	74.7	+3.4
+ all rules	**82.2**	**74.7**	**78.3**	**+7.0**
Automatic QA-SRL				
HUM-QQ	71.1	79.0	74.8	+3.5
– conn. pen.	71.6	75.7	73.6	+2.3
+ lex	**79.8**	**83.4**	**81.6**	**+10.3**
+ lex + MI	77.7	82.1	79.9	+8.6

Table 10: B^3 Results on models we tested. The gap between HUM-QQ and SYNTF is larger than for purity and collocation, as B^3 is a tougher metric which is more discriminative between clusterings. The last model variant (+MI) is described in Appendix G.

G Related Clustering Algorithms

Recall the Hard Unigram Mixture loss

$$\mathcal{L}_\lambda^{\text{HUM}}(\mathbf{C}) = -\log \mathrm{P}(\mathbf{X} \mid \mathbf{C}) - \lambda \log \mathrm{P}(\mathbf{C}).$$

Different settings of λ reproduce several objectives present in the literature, summarized in Table 9. As written in Section 3, when $\lambda = 1$, minimizing $\mathcal{L}_1^{\text{HUM}}$ maximizes likelihood of the data $\mathbf{X}$ under a mixture of unigrams model (Nigam et al., 2000).

When the number of clusters k is fixed, setting $\lambda = 0$ as in our greedy merging step (Section 3.3) is equivalent to enforcing a uniform prior π over mixture components. In this case, the gain in loss on each merge is the Jensen-Shannon Divergence (JSD) between the merged clusters, scaled by their total size and using each cluster's size to determine its mixing weights in the divergence, as in the mixture-based definition of JSD by Lin (1991). JSD is used in the same way by Chrupała (2012), without the scaling and weighting, as a similarity measure for agglomerative clustering.

Finally, setting $\lambda = -1$ reduces the HUM loss to the mutual information between the QA-SRL questions under ϕ and the cluster assignment C, which has been used in prior work to encourage informative clusterings (Michael et al., 2020). This is related to the *distributional clustering* paradigm of Pereira et al. (1993), which aims to identify common factors that explain distributional data, and which Slonim and Tishby (1999) frame in terms of an information bottleneck that maximizes mutual information between the data and a jointly distributed 'relevance' variable (though in our case, the reference variable is the cluster assignment itself). Setting $\lambda = -1$ in the greedy merging step, we find (in Table 10) that using a mutual information criterion in this way hurts performance. We guess this is because the objective incentivizes clusters of uniform size, which does not match the highly skewed distributions of gold semantic roles.

H Manual Analysis Results

H.1 Improved Verbs on A0

The top 50 verbs by F1 gain on A0 from SYNTF to HUM-QQ are: *compete*, *conduct*, *connect*, *combine*, *dominate*, *restore*, *require*, *yield*, *limit*, *ban*, *direct*, *tie*, *oversee*, *contain*, *identify*, *increase*, *evaluate*, *specialize*, *allow*, *assist*, *restrict*, *found*, *grant*, *feature*, *propose*, *detail*, *force*, *convert*, *veto*, *rate*, *bolster*, *appoint*, *enact*, *design*, *list*, *lead*, *resolve*, *retire*, *schedule*, *reach*, *analyze*, *remove*, *speed*, *manage*, *deliver*, *underlie*, *revise*, *emerge*, *enable*, *block*.

We examined 30 sentences containing the top 3 verbs (*compete*, *conduct*, and *connect*). There were 31 A0 arguments of these verbs in these sentences. Of these, 8 (26%) were overt, 11 (35%) were extracted subjects of relative clauses, 5 (16%) were modified by the predicate appearing in an adjectival clause, 5 (16%) were subjects of open complements of control verbs, and 2 (6%) were otherwise implicit (subject of an adverbial clause or open complement not under a control verb).

H.2 Improved Verbs on A1

We examined the top 50 verbs by their difference in B^3 performance on A1 between SYNTF and HUM-QQ. 48 of them are transitive; the other two are bolded. In decreasing order of F1 gain, they are: *propose*, *prefer*, *price*, *relate*, *involve*, *help*, *choose*, *consider*, *design*, *mention*, *identify*, *release*, *include*, ***exist***, ***range***, *value*, *revise*, *lead*, *associate*, *need*, *increase*, *import*, *prove*, *feel*, *place*, *determine*, *limit*, *found*, *enact*, *control*, *cancel*, *dilute*, *disclose*, *select*, *exclude*, *force*, *insure*, *accrue*, *damage*, *calculate*, *hurt*, *secure*, *delay*, *regard*, *record*, *open*, *use*, *concern*, *weaken*, *adjust*.

I Calculating Normalized PMI

Here we describe some special concerns for our use of normalized PMI in Section 6.2.

Pointwise mutual information (PMI) is a measure of how likely two items (such as tokens in a corpus) are to occur together relative to chance (Church and Hanks, 1989). One feature of PMI is that it tends to be larger for rare events: if two items x and y always occur together, then their PMI is $-\log \mathrm{P}(x, y)$. This can make it difficult to assess association patterns among items with greatly varying probabilities (e.g., the AM-CAU role appears for 1% of arguments, while A1 appears for 27%). So we use *normalized* PMI (NPMI; Bouma, 2009), which factors out the effect of item frequency on PMI. Formally, the NPMI of x and y is

$$\left(\log \frac{\mathrm{P}(x, y)}{\mathrm{P}(x)\,\mathrm{P}(y)}\right) \Big/ -\log(\mathrm{P}(x, y)) \;, \quad (1)$$

taking the limit value of -1 when they never occur together, 1 when they only occur together, and 0 when they occur independently. We use NPMI to analyze the co-occurrence of *gold labels* in *predicted clusters*: A pair of gold labels with high NPMI are preferentially grouped together by the induced roleset, whereas two labels with low NPMI are preferentially distinguished. The joint distribution between gold labels is generated by drawing one point (x) uniformly at random from the data, drawing another (y) uniformly at random from x's predicted cluster, and reading the gold labels of both. NPMI has been used to analyze clusters in this way by Michael et al. (2020).

Calculating NPMI naïvely on our full clustering has a caveat. The denominator of the PMI term in Equation 1, $\mathrm{P}(x)\,\mathrm{P}(y)$, uses marginal probabilities of x and y over the corpus to calculate chance co-occurrence. But our clusters are constrained not to overlap between verbs, so this does not correctly estimate chance cooccurrence in our setting. Instead, we use the expectation over verbs of within-verb chance cooccurrence:

$$\sum_{v} \mathrm{P}(x \mid v)\,\mathrm{P}(y \mid v)\,\mathrm{P}(v),$$

where $\mathrm{P}(v)$ is proportional to the number of arguments for the verb v.

J Question Distributions by Role

We list the top questions and their probabilities for modifier roles in Table 11. Questions for core roles and the ones covered by our lexical rules are in Table 12. We use *verb* (or *verbs*, or *verbed*) as a placeholder for the verb, which in practice is replaced with the predicate for a given instance.

Role	Top Questions	Prob
TMP	When does something verb something?	0.34
	When does something verb?	0.21
	When is something verbed?	0.18
	When does something verb somewhere?	0.03
	When does sth. verb to do something?	0.02
	How does something verb?	0.01
	How is something verbed?	0.01
ADV	Why does something verb something?	0.13
	How does something verb something?	0.12
	When does something verb something?	0.09
	How is something verbed?	0.08
	How does something verb?	0.08
	Why does something verb?	0.05
	Why is something verbed?	0.04
	When does something verb?	0.04
	What does something verb?	0.03
	When is something verbed?	0.03
MNR	How is something verbed?	0.25
	How does something verb?	0.22
	How does something verb something?	0.19
	What does something verb?	0.02
	Where does something verb?	0.02
	Why does something verb something?	0.02
	How does something verb somewhere?	0.02
LOC	Where does something verb something?	0.24
	Where is something verbed?	0.22
	Where does something verb?	0.21
	When does something verb something?	0.04
	How does something verb something?	0.03
	How does something verb?	0.02
	How is something verbed?	0.02
PNC	Why does something verb something?	0.29
	Why is something verbed?	0.21
	Why does something verb?	0.08
	Why does something verb somewhere?	0.05
	What is something verbed to do?	0.03
	How is something verbed?	0.03
	What is something verbed for?	0.02
CAU	Why does something verb something?	0.32
	Why does something verb?	0.16
	Why is something verbed?	0.16
	Why does something verb somewhere?	0.04
	How does something verb?	0.04
	Why does sth. verb to do something?	0.03
	How does something verb something?	0.02
DIR	Where does something verb?	0.40
	How does something verb?	0.17
	Where is something verbed?	0.10
	Where does something verb something?	0.07
	How is something verbed?	0.06
	How does something verb something?	0.03

Table 11: Top questions in the QA-SRL features on the training set for modifier roles. Most of the roles align with a particular *wh*-word especially well, especially for *when*, *where*, and *why*. But AM-ADV takes a variety of *wh*-words, and *how* appears often for nearly all modifier roles. In longer questions, 'something' is abbreviated for space.

Role	Top Questions	Prob
A0	What verbs something?	.65
	What verbs?	.14
	How is something verbed?	.02
A1	What does something verb?	.42
	What is verbed?	.25
	What verbs?	.09
	What verbs something?	.03
	What does something verb to do?	.02
A2	What does something verb?	.12
	How is something verbed?	.07
	What verbs something?	.07
	Where is something verbed?	.06
	What is verbed?	.06
	How does something verb?	.06
	How much does something verb?	.04
A3	How does something verb?	.15
	What does something verb?	.09
	How is something verbed?	.07
	Why does something verb something?	.06
	How does something verb something?	.05
	When does something verb?	.05
	Where does something verb?	.04
A4	What does something verb to?	.17
	Where does something verb?	.17
	How does something verb?	.16
	How much does something verb?	.14
	What does something verb something to?	.04
	How is something verbed?	.03
NEG	What verbs something?	.40
	What verbs?	.15
	What is verbed?	.12
	How is something verbed?	.05
	How does something verb?	.03
	How does something verb something?	.03
MOD	What verbs something?	.22
	How does something verb something?	.11
	What verbs?	.09
	Why does something verb something?	.07
	What is verbed?	.06
	How does something verb?	.06
	How is something verbed?	.06
DIS	When does something verb something?	.15
	How does something verb something?	.15
	What verbs something?	.08
	How is something verbed?	.07
	How does something verb?	.07
	Why does something verb something?	.06
	When does something verb?	.05

Table 12: Top questions in the QA-SRL features on the training set for core roles and the ones covered by our lexical rules. The questions for AM−NEG, AM−MOD, and AM−DIS often don't make sense, e.g., asking for the subject of the verb. No QA-SRL questions are appropriate or were annotated for many arguments of these types. On the other hand, the core roles behave essentially as expected: A0 is dominated by the subject, A1 has a mix of subjects and objects, with some complements, and A2 and on have a wider spread of different expressions. Since the core argument roles have predicate-specific meanings, the distributions here can only be interpreted as aggregates across many such meanings.