LaSQuE: Improved Zero-Shot Classification from Explanations Through Quantifier Modeling and Curriculum Learning

Sayan Ghosh[*] **Rakesh R Menon**[*] **Shashank Srivastava**
UNC Chapel Hill
{sayghosh, rrmenon, ssrivastava}@cs.unc.edu

Abstract

A hallmark of human intelligence is the ability to learn new concepts purely from language. Several recent approaches have explored training machine learning models via natural language supervision. However, these approaches fall short in leveraging linguistic quantifiers (such as 'always' or 'rarely') and mimicking humans in compositionally learning complex tasks. Here, we present LaSQuE, a method that can learn zero-shot classifiers from language explanations by using three new strategies - (1) modeling the semantics of linguistic quantifiers in explanations (including exploiting ordinal strength relationships, such as 'always' > 'likely'), (2) aggregating information from multiple explanations using an attention-based mechanism, and (3) model training via curriculum learning. With these strategies, LaSQuE outperforms prior work, showing an absolute gain of up to 7% in generalizing to unseen real-world classification tasks.[1]

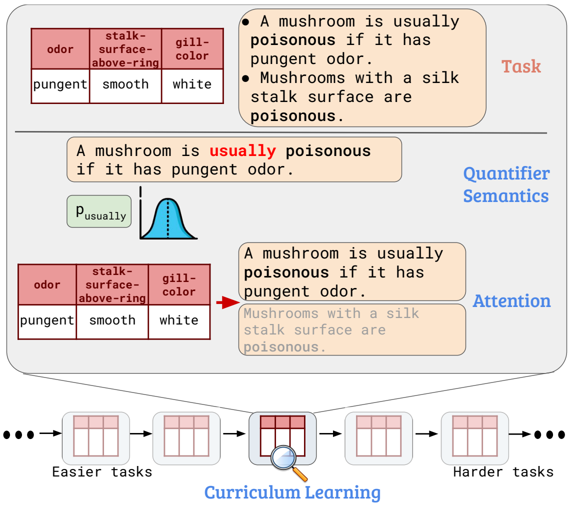

odor
stalk-surface-above-ring
gill-color
pungent
smooth
white
• A mushroom is usually poisonous if it has pungent odor.
• Mushrooms with a silk stalk surface are poisonous.
Task
A mushroom is usually poisonous if it has pungent odor.
P usually
Quantifier Semantics
odor
stalk-surface-above-ring
gill-color
pungent
smooth
white
A mushroom is usually poisonous if it has pungent odor.
Mushrooms with a silk stalk surface are poisonous.
Attention
Easier tasks
Harder tasks
Curriculum Learning

Figure 1: We present techniques to learn zero-shot classifiers from natural language explanations. We investigate (1) learning Quantifier Semantics to fully leverage supervision from individual explanations, (2) Attention-based mechanisms to identify the most salient explanations for learning a task, and (3) Curriculum Learning to learn more complex tasks progressively.

1 Introduction

Learning from language (also 'conversational machine learning') is a new paradigm of machine learning where machines are taught tasks through natural language supervision in the form of explanations and instructions (Andreas et al., 2018; Arabshahi et al., 2020; Weller et al., 2020; Efrat and Levy, 2020). Language explanations of concepts have been explored for training classification models in few-shot and zero-shot settings (Mei et al., 2022; Srivastava et al., 2017, 2018; Hancock et al., 2018; Chai et al., 2020; Obeidat et al., 2019; Hanjie et al., 2022).

However, current approaches fall short in fully leveraging supervision available in language explanations and using learning strategies that humans routinely employ in learning new tasks. First, most approaches, such as LNL (Srivastava et al., 2017), and BabbleLabble (Hancock et al., 2018), do not model supervision within free-form language explanations in the form of quantifiers. Quantifiers are linguistic elements that can dictate the vagueness and perceived confidence of relations expressed in a statement (Solt, 2009; Moxey and Sanford, 1986). For example, with statements such as *'some poisonous mushrooms are red in color'*, we can infer that a red mushroom is not always poisonous because of the quantifier *'some'*. Moreover, quantifiers are a ubiquitous part of natural language and universal across languages. Second, prior approaches do not reason about differences in salience and utility of multiple explanations in learning a new task, weighing them equally in the absence of labeled data. This is sub-optimal since certain explanations can be naturally harder to incorporate or have inherently less value in learning

[*] Equal contribution

[1] Our code can be found at: https://github.com/sgdgp/LaSQuE

Findings of the Association for Computational Linguistics: ACL 2023, pages 7403–7419
July 9-14, 2023 ©2023 Association for Computational Linguistics

a concept[2]. Thirdly, when learning a set of tasks, humans often learn 'simpler' concepts first and gradually build towards 'harder' concepts (Newport, 1990). Curriculum learning (Bengio et al., 2009), a method where tasks are introduced in an incremental and adaptive manner, has been shown to be effective in a wide range of complex machine learning tasks (Platanios et al., 2019; Tay et al., 2019; Narvekar et al., 2017). However, its application in the context of learning from explanations has yet to be explored. The deteriorating generalization of classifiers with the increasing complexity of explanations in prior work (Menon et al., 2022) further motivates the need for curriculum learning for learning from explanations.

To address the first shortcoming, our approach LaSQuE (**Lea**rning **S**trategies For **Qu**antified **E**xplanations) explicitly models quantifier semantics and learns them directly from labeled classification data. However, directly learning from labeled data can lead to quantifier semantics that is inconsistent with human perceptions of their numerical estimates. Hence, we provide weak supervision in the form of ordinal relations describing the relative strengths of quantifiers (e.g., 'always' > 'likely') to supplement the learning of quantifier semantics that comply with human judgments. Second, we design an attention-based mechanism to model the relative importance of multiple explanations in classifying an example. We also qualitatively analyze the attention weights to identify characteristics of explanations found most helpful. Finally, we consider different axes of explanation complexity and empirically evaluate the utility of curriculum learning on three different curricula.
As our test bed, we use the recently proposed CLUES benchmark (Menon et al., 2022) for learning classification tasks from language explanations. Our work focuses on learning classifiers from language explanations where the explanations provide the logic to perform the classification. (e.g., the explanation 'pungent mushrooms are toxic', provides the logic that mushrooms with a pungent odor should be classified as toxic). CLUES is the largest available benchmark that contains explanations conformant with this perspective. It differs from some other benchmarks (Mishra et al., 2022; Sanh et al., 2022), where the language component provides the *description of the task* instead (such as, 'classify the mushrooms as toxic or poisonous'), which can be used to train/prompt a model. On CLUES, LaSQuE achieves an improvement of 17% and 7%, respectively, on the synthetic and real-world benchmarks over baselines.

The rest of this paper is structured as follows: we provide a description of the preliminaries in §3. In §4 we describe LaSQuE and our learning strategies along with supporting empirical performance. §5 discusses performance of LaSQuE on real world classification tasks. Our contributions are:

- We introduce LaSQuE, which models semantics of linguistic quantifiers, and uses an attention-based mechanism to identify salient explanations for learning classifiers from language. LaSQuE significantly outperforms previous methods in generalizing to unseen classification tasks.
- We empirically demonstrate the utility of curriculum learning in training classifiers from language by experimenting with three curricula.

2 Related Work

Natural Language Quantification. Previous work has studied the role of quantifiers in natural language from multiple perspectives, such as formal logic (Barwise and Cooper, 1981), linguistics (Lobner, 1986; Bach et al., 2013), cognitive psychology (Kurtzman and MacDonald, 1993), and natural language processing to guide statistical models (Srivastava et al., 2018). In the above mentioned works, quantifiers have been typically modelled in either set-theoretic terms (Barwise and Cooper, 1981) or by representing them probabilistically (Moxey and Sanford, 1993; Yildirim et al., 2013; Srivastava et al., 2018). Our work is closely related to Srivastava et al. (2018), who also model the effects of quantifiers in modifying the belief of a classifier. However, we differ from Srivastava et al. (2018) as we learn the beliefs associated with quantifiers during training as opposed to defining them apriori with fixed values. More recently, Cui et al. (2022) discusses the challenges in understanding quantifiers, specifically in the context of NLI, and contributes a focused test dataset to benchmark NLI models on their ability to understand quantifiers. While both Cui et al. (2022) and our work broadly highlight the need to model quantifiers to advance language understanding, we differ in the nature of downstream tasks studied (diverse classification tasks in our work vs NLI in Cui et al. (2022)). Our approach (LaSQuE) contains a dedicated module that enables us to *learn* quantifier

[2] e.g., overly complex or highly subjective explanations

semantics, which apply to a wide range of tasks spanning multiple domains.

Curriculum Learning. Curriculum learning (Bengio et al., 2009) is a technique to learn complex tasks through a graded exposure of examples ranging from easy-to-hard difficulty. Recent works in machine learning (Jiang et al., 2018; Guo et al., 2018; Hacohen and Weinshall, 2019) have successfully demonstrated the utility of curriculum learning in learning image classification tasks. More recently, Xu et al. (2020) also demonstrated the effectiveness of curriculum learning for a set of natural language understanding tasks drawn from the GLUE benchmark (Wang et al., 2018). However, prior works build a curriculum of easy-to-hard examples to improve model performance on individual tasks. Rather than examples, we build a curriculum of easy-to-hard tasks in our work, similar to Mao et al. (2019). In contrast to Mao et al. (2019) though, we focus on learning structured data classification tasks from language explanations as opposed to visual question answering tasks.

3 Preliminaries

3.1 Setup

We employ a cross-task generalization setup (Mishra et al., 2022), and train classifiers using multi-task training over a set of tasks $\mathcal{T}_{seen}$ and evaluate for zero-shot generalization on a set of tasks $\mathcal{T}_{novel}$ ($\mathcal{T}_{novel} \cap \mathcal{T}_{seen} = \phi$). The evaluation metric is the zero-shot classification accuracy on novel classification tasks.

Datasets. For experiments, we use the recently proposed CLUES benchmark (Menon et al., 2022). The benchmark is composed of synthetic and real-world classification datasets. In CLUES, inputs are structured, consisting of attribute name-attribute value pairs (see Figure 1 for example). We use the 'Features-as-Text' or 'FaT' representation to encode the examples following Menon et al. (2022), i.e., given the input as in Figure 1, we encode the input as text tokens in the form `odor | pungent [SEP] ...gill-color | white [SEP]`. Additional details and statistics about CLUES can be found in Appendix A.

Baselines. To compare the efficacy of our proposed strategies on CLUES, we use the following two baselines in our experiments: (1) RoBERTa w/o Exp (does not use explanations) (Liu et al., 2019) and (2) ExEnt (Menon et al., 2022). ExEnt uses Natural Language Inference (NLI) as an intermediate step to perform classification. The operations in ExEnt can be broadly grouped into three steps: (1) *NLI step*: obtain scores from an entailment prediction model (RoBERTa+MNLI-finetuned) for the alignment between the input and each explanation available for a task; (2) *Entailment → Classification scores conversion*: convert the entailment scores for each input-explanation pair into classification scores based on the nature of the explanation; and (3) *Aggregation*: average the classification scores from each input-explanation pair to obtain an aggregate score for classification. Convert aggregate scores to probabilities using softmax and train the model end-to-end using the cross-entropy loss. For more details on ExEnt, we refer the reader to Menon et al. (2022).

4 LaSQuE

In this section, we present our method, LaSQuE, and provide detailed descriptions and empirical support for the different learning strategies that are part of LaSQuE– (1) modeling quantifier semantics, (2) using attention for aggregation across explanations, and (3) curriculum learning.

4.1 Modeling quantifier semantics

Quantifiers are a ubiquitous part of natural language and can help express varying strengths of relations in a statement. Prior work in cognitive science (Chopra et al., 2019; Steinert-Threlkeld, 2021) and machine learning (Srivastava et al., 2018; Menon et al., 2022) shows that people tend to use quantifiers often in learning or teaching tasks. Hence, modeling quantifiers is important for building systems that can mimic humans in efficiently learning from natural language. However, past work on computational modeling of quantifiers is sparse. To the best of our knowledge, no prior work has explored learning quantifier semantics in a data-driven way. In this work, we devise methods to explicitly model the differential semantics of quantifiers present in explanations to guide classifier training. Figure 2 shows architecture of our model, LaSQuE.

To formalize our approach to modeling quantifier semantics, consider a task t with the set of class labels L and set of explanations E. Given the Feature-as-Text (FaT) representation of a structured data example $x \in t$ and an explanation

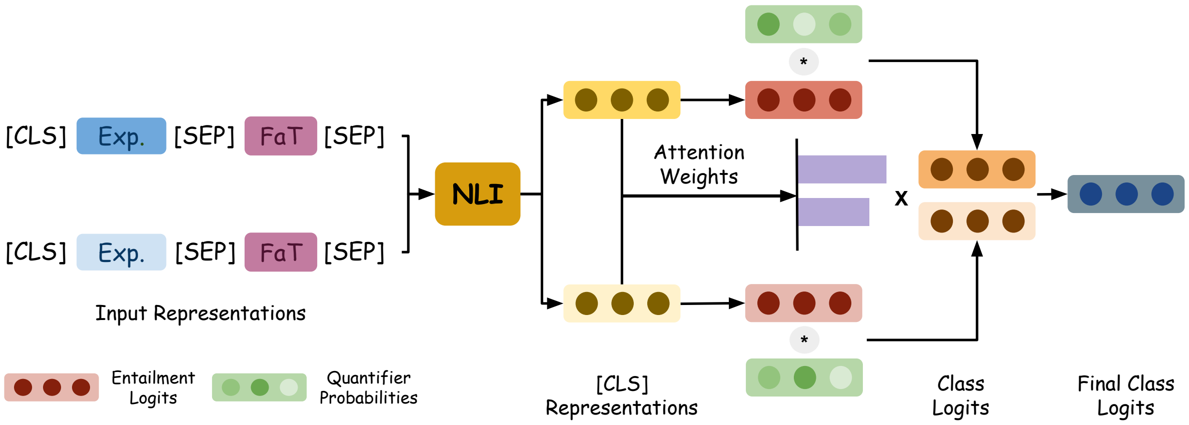


Figure 2: LaSQuE models quantifier semantics and uses attention over multiple explanations to aggregate class logits. We re-weigh the logits from the NLI step and map them to class logits, thus strengthening/weakening the contribution of an explanation towards assigning the label (mentioned in the explanation) to the input. This mapping procedure (described in §4.1) also takes the strength of the quantifier (in form of probabilities) present in the explanation into account (denoted by ⊛ in the figure). Curriculum learning (not shown in the figure) entails training LaSQuE progressively on easy-to-hard tasks.

$e_j \in E$, our model takes FaT(x) and e_j as input and passes it through a pretrained RoBERTa+MNLI model, following previous work (Menon et al., 2022). For each example-explanation pair, the NLI model outputs entailment, neutral, and contradiction scores (denoted as s_e^j, s_n^j, and s_c^j respectively). In the next step, we incorporate quantifier semantics to assign logits to the set of class labels, L, using the outputs of the NLI model. In this work, we model the semantics of a quantifier by a probability value signifying the strength of the quantifier, i.e., the confidence of the quantifier in conveying the beliefs expressed in the explanation. Then the class logit assignment is done as follows. If:

- **Explanation e_j mentions a label l_{exp}**: An illustrative example is 'If head equal to 1, then it is usually dax'. In this example the label mentioned, l_{exp} is 'dax'. Let p_{quant} denote the strength (as a probability) of the quantifier mentioned in the explanation[3]. In the aforementioned example, p_{quant} will be the probability associated with the quantifier 'usually'. Let $\mathbb{P}(l)$ denote the probability of any label $l \in L$. Then,

$$\log(\mathbb{P}(l_{exp})) \propto p_{quant} \times s_e^j + (1 - p_{quant}) \times s_c^j + s_n^j/|L| \quad (1)$$

$\forall\, l \in L \setminus \{l_{exp}\}$,

$$\log(\mathbb{P}(l)) \propto p_{quant} \times s_c^j + (1 - p_{quant}) \times s_e^j + s_n^j/|L| \quad (2)$$

Equations 1 and 2 define the likelihood of each label in terms of the NLI model outputs. The entailment score (s_e) denotes how strongly the explanation influences the classifier to label the input as l_{exp}. On the other hand, the contradiction score, s_c denotes how strongly the explanation influences the classifier to *not* label the input as l_{exp}. These 'influences' are additionally modified based on the quantifier strength as shown in equations 1 and 2.
Note: If quantifiers are absent in the explanations, we assume p_{quant} is 1.

- **Explanation e_j mentions negation of a label 'l_{exp}' (NOT l_{exp})**: An illustrative example is 'If head equal to 1, then it is usually not dax', where 'dax' is the label mentioned (l_{exp}). The roles of s_c^j and s_e^j as described in the previous equations are reversed.

Following this step, we average the class logits from each example-explanation pair to aggregate the decisions. Finally, we apply a softmax over the resulting class scores to obtain a distribution over class labels and train the model to minimize the cross-entropy loss, $\mathcal{L}_{CE}$.

Approaches to learn quantifier semantics. We experiment with the following approaches to learn the probability values of quantifiers:

[3] We assume that explanations contain a single quantifier. This assumption also holds true with the explanations found in CLUES.

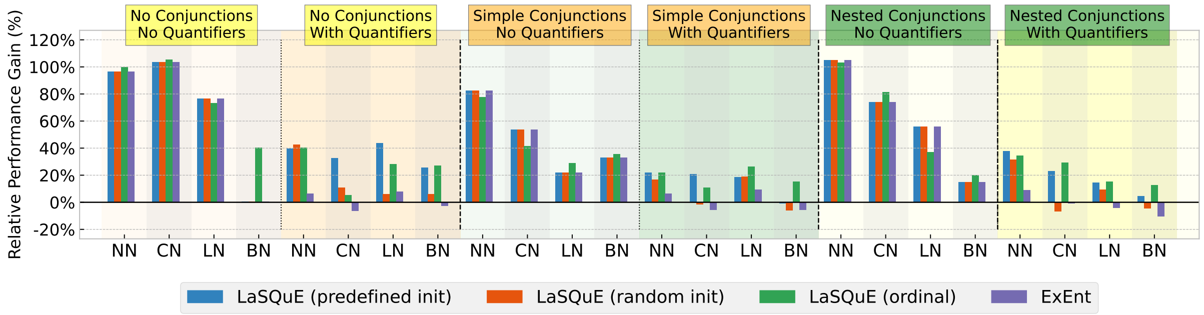


Figure 3: Relative performance gain of LaSQuE and ExEnt with respect to the baseline RoBERTa w/o Exp. model. 'NN', 'CN', 'LN', and 'BN' stand for 'no negations', 'clause negations', 'label negations', and 'both negations' respectively denoting the variations of negations appearing in the explanations of CLUES-Synthetic. We see that LaSQuE outperforms ExEnt (Menon et al., 2022) on all tasks having explanations with quantifiers. Within each panel, the complexity of tasks increases from left to right due to negations. Across panels, the complexities also increase from left to right due to a combination of change in the presence of quantifiers and explanation structure (using conjunctions/disjunctions).

- Finetuning pre-defined probability values: We initialize the quantifier probability values (p_{quant}) with pre-defined values and fine-tune them while training LaSQuE. These initial estimates can be specified from domain knowledge or by an expert. In this work, we adopt the quantifier values from Srivastava et al. (2018).[4] We refer to the model learned using this approach as LaSQuE (predefined init).
- Learning probability values for the quantifiers from scratch: We start from random initialization and then learn the probability values of each quantifier while training LaSQuE. We refer to the model learned using this approach as LaSQuE (random init).
- Ordinal ranking as weak supervision: We explore another form of supervision by specifying ordinal relationships between pairs of quantifiers based on their relative strengths. To define ordinal relationships, we re-purpose the quantifier probability values in Srivastava et al. (2018). For example, quantifiers such as 'likely' and 'often' associated with the values 0.7 and 0.5 respectively are defined by the relationship, 'likely' > 'often'. We leverage the ordinal relations to guide the learning of quantifier semantics through a ranking loss, following Pavlakos et al. (2018). Given a pair of quantifiers q_i and q_j $(i \neq j)$, the ranking loss is defined as:

$$\mathcal{L}_{i,j} = \begin{cases} \log(1 + \exp(p_{q_i} - p_{q_j})), & \mathbf{p^*_{q_i}} > \mathbf{p^*_{q_j}} \\ (p_{q_i} - p_{q_j})^2, & \mathbf{p^*_{q_i}} = \mathbf{p^*_{q_j}} \end{cases}$$

where, $\mathbf{p^*_q}$ refers to the subjective probability value of a quantifier, q, in Srivastava et al. (2018). Further, we define

$$\mathcal{L}_{rank} = \sum_{(q_i, q_j) \in Q} \mathcal{L}_{i,j} \tag{3}$$

where, Q denotes the full set of quantifiers present in the explanations of CLUES (§A.1). The final loss is a weighted sum of classification loss ($\mathcal{L}_{CE}$) and ranking loss ($\mathcal{L}_{rank}$).

$$\mathcal{L}_{total} = \mathcal{L}_{CE} + \lambda \mathcal{L}_{rank} \tag{4}$$

where, λ denotes the weight of ranking loss. We use $\lambda = 10$ in this work, chosen using validation performance. We refer to the model learned using this approach as LaSQuE (ordinal).

Performance on CLUES-Synthetic: To evaluate the effectiveness of natural language quantification in learning classifiers from language explanations, we experiment on a collection of 100 tasks for each of the 48 different complexities from CLUES-Synthetic. The complexities vary based on the presence of conjunctions, negations, and quantifiers in the task explanations. For each complexity, we train a classifier and evaluate its generalization to novel tasks of the same complexity.

Figure 3 shows the results of different variants of LaSQuE and ExEnt across the different task complexities as the relative performance gain over the RoBERTa w/o Exp. baseline for zero-shot classification of examples from unseen tasks. For ease of visualization, we have averaged the results across binary and multiclass classification tasks in the

[4] Full list of quantifiers used can be found in the Appendix.

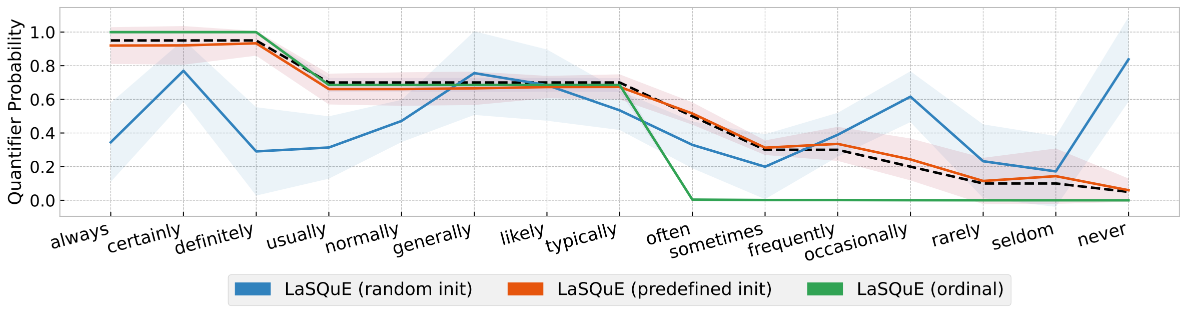


Figure 4: Quantifier probability values learned by the different approaches mentioned in §4.1. The solid line and the shaded region denote the mean and standard deviation respectively of the learned probabilities for a given approach across 48 synthetic task complexities in `CLUES-Synthetic`. The dotted-line denotes (1) probability values used by Menon et al. (2022) to create synthetic tasks of `CLUES-Synthetic` and (2) the quantifier probability initialization values for `LaSQuE` (predefined init).

figure. Post-averaging, we plot sets of four bars corresponding to the evaluations of the four models (three `LaSQuE` variations + `ExEnt`) on each of the 24 task complexities resulting from negations, conjunctions, and quantifiers.

Overall, we find that explicit modeling of quantifier semantics helps to learn better zero-shot classifiers. In particular, `LaSQuE` expectedly performs much better than previous approaches on tasks with quantified explanations. Further, while `ExEnt` is weaker than RoBERTa w/o exp. baseline on certain task complexities, `LaSQuE` outperforms or match the baselines on almost all task complexities. Expectedly, the generalization ability of models decrease with the increasing complexity of explanations due to changes in the structure of explanations or the presence of negations.

METHOD	ACCURACY (↑)
`ExEnt`	54.7
`LaSQuE` (random init)	56.9♦
`LaSQuE` (predefined init)	59.7♦♣
`LaSQuE` (ordinal)	59.9♦♣

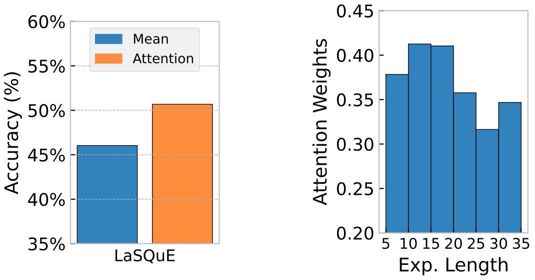
Table 1: Average accuracies of different models on `CLUES-Synthetic`. ♦ and ♣ denote that the model is significantly better than `ExEnt` and `LaSQuE` (random init) respectively. For all significance tests $p < 0.005$ using a paired t-test.

Table 1 shows the average accuracy of different `LaSQuE` variants and `ExEnt` over tasks in `CLUES-Synthetic`. `LaSQuE` (ordinal) performs the best across majority of the synthetic task complexities in `CLUES` with a significant 5.2% absolute improvement across all tasks complexities over `ExEnt`. Further, `LaSQuE` (predefined init) performs comparably with `LaSQuE` (ordinal) in many cases (5.0% vs 5.2% absolute improvement over `ExEnt`) but struggles in tasks where explanations have negations in both clauses and labels. The poor performance of `LaSQuE` (random init) compared to `LaSQuE` (predefined init) and `LaSQuE` (ordinal) demonstrates the challenge of jointly learning quantifier semantics and a classifier only from labels. Nevertheless, `LaSQuE` (random init) outperforms `ExEnt` significantly by 2.2% points (absolute) on average across all synthetic task complexities.

Analyzing the learned quantifier estimates for the `LaSQuE` variant whose quantifier values are fine-tuned from predefined values (`LaSQuE` (predefined init) in Figure 4), we observe the final learned probability values are close to the initialization values. On the other hand, we note that `LaSQuE` (ordinal) learns three clusters of quantifier probabilities that match with our intuition of high-strength (probability above 0.95), intermediate strength (probability around 0.7), and low-strength quantifiers (probability close to 0). Even though `LaSQuE` (ordinal) makes little difference between quantifiers within a cluster, we observe that weak supervision in the form of ordinal ranks is sufficient to develop models competent with, even surpassing, `LaSQuE` (predefined) that uses predefined initialization. Finally, we observe `LaSQuE` (random init) struggles to learn any interpretable ranking for quantifiers. On further analysis, we identify that `LaSQuE` (random init) can learn the quantifier semantics reasonably well for simple binary tasks. However, it struggles to learn reasonable quantifier semantics in the presence of negations, conjunctions, and disjunctions.

4.2 Aggregating explanations with attention

To mimic human learning, models need to identify salient explanations that can be potentially useful in classifying an input. As previously mentioned, in the absence of labeled data for a task, previous work on learning from explanations does not differentiate between multiple explanations in terms of their salience and utility for classifying an example. For example, `ExEnt` averages the class logits from multiple explanations for making predictions, implicitly considering all explanations equally salient for classifying an example. To model the varying importance of each explanation towards deciding the class label, we use attention for the aggregation step. We obtain the attention weights by using a feed-forward network over the `[CLS]` representations obtained from the intermediate NLI model. The attention weights are then normalized using softmax. The final aggregated class logits for the label l is $\sum_{j=1}^{m} a_j z_j^l$, where a_j is the attention weight for each explanation e_j, and z_j^l denotes the logit for label l using e_j. The aggregated class logits are converted to probabilities using softmax, and the model is trained using cross-entropy loss.



(a)

(b)

Figure 5: (a) Generalization accuracy on `CLUES-Synthetic` ablating the use of attention to combine results from multiple explanations in `LaSQuE`. (b) Mean attention scores of explanations from `LaSQuE` vs explanation length (# of tokens).

Performance on `CLUES-Synthetic`: To evaluate the role of attention, we experiment with two models, one using mean and the other using attention for aggregation. Each model is fine-tuned from the RoBERTA+MNLI backbone on the training tasks of `CLUES-Synthetic`. Figure 5(a) shows the generalization performance for two variants of `LaSQuE`. Using attention for aggregation across explanations results in significantly better generalization accuracy (50.68% vs 46.04% ; $p < 0.1$, paired t-test). While technically simple, we see that this modification allows the model to behave in conceptually sophisticated ways.

Attention weight analysis: Figure 5(b) shows a histogram of average attention weights from `LaSQuE` for different explanation lengths. We find that longer explanations (typically explanations with nested conjunctions and disjunctions) get lower attention weights on average. This seems reasonable and intuitive since complex explanations are likely harder for the model to interpret correctly, so relying on them may be riskier. Further, we find that explanations containing quantifiers receive higher attention on average than explanations without quantifiers (0.44 vs 0.35), further highlighting the value of modeling quantifiers in explanations. Explanations containing ‘definitely’ and ‘frequently’ received higher attention than explanations containing other quantifiers. Surprisingly, the average attention weights were comparable for explanations with and without negation.

4.3 Curriculum learning

From Figure 3, it is clear that the generalization abilities of models diminish dramatically with the increasing complexity of tasks and explanations. Thus, we next investigate using curriculum learning (Bengio et al., 2009), which has shown significant successes in learning complex tasks, for learning classifiers from explanations.

We define the ‘complexity’ of an explanation under three axes here - (1) the type of classification task (binary vs multiclass) served by the explanation, (2) presence of negations in the explanation, and (3) structure of the explanation (whether the explanation contains conjunction/disjunctions or nested clauses). Using curriculum learning we empirically evaluate if training on a classification task with ‘easier’ (less complex) explanations first gives any advantage when learning a task with ‘harder’ (more complex) explanations. In this work, we explore the following curricula:

- Binary → multiclass: We first train classifiers on binary classification tasks and then on multiclass classification tasks.
- No negations → having negations in labels and clauses: We train on tasks with explanations that contain no negation followed by training on tasks with explanations that have negations in them. Note that negation can appear in the clauses or before a class label in the explanation.
- No conjunctions/disjunctions → tasks with

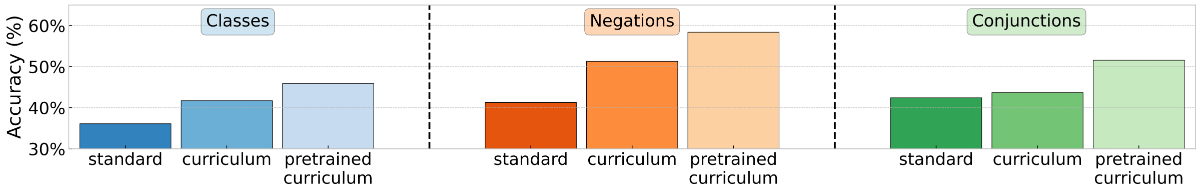


Figure 6: Averaged generalization accuracy on novel classification tasks of the 'most difficult' task complexities across three curricula: classes, negations, conjunctions. While effective in all three curricula, curriculum learning shows maximum gains when handling negations.

nested conjunctions and disjunctions: We first train on tasks with simple explanations without any conjunctions or disjunctions. Following this, we train on tasks having explanations that contain one conjunction/disjunction and then, on tasks with explanations that contain nested clauses.

Figure 6 shows the results of curriculum learning on the synthetic tasks of CLUES. We find that LaSQuE trained through curriculum learning (denoted by 'curriculum' bars) outperforms LaSQuE trained only on the most challenging task set in the corresponding curriculum (denoted by 'standard' bars) on the generalization accuracy of novel hardest tasks in the corresponding curriculum. However, we notice that LaSQuE has minimal benefits from training in a curriculum learning fashion on the conjunctions curriculum (shown in green).

We hypothesize that jointly learning quantifiers and classifiers might be challenging, so we experiment with another setup where we reduce the learning problem to only modeling task complexities by freezing the quantifier semantics with the semantics learned by LaSQuE on simple synthetic binary tasks. With this modification, we find that curriculum learning is much more effective in all three curricula as seen from the improved average generalization performance (denoted by 'pretrained curricula' bars in Figure 6). Notably, we find curriculum learning to be most effective in handling negations obtaining an absolute improvement of 17.11% on the generalization accuracy. The low gains achieved through curriculum learning for handling structural complexity indicates a need to model the role of conjunctions and disjunctions explicitly. We leave this for future work to explore.

We further analyze the progression of the zero-shot generalization accuracies as we increase the complexity of tasks as we move forward in the curriculum. We defer this result and discussion to the Appendix §C. Briefly, our results suggest that models tend to perform better on more complex tasks at the expense of slight performance drops on simpler tasks as the curriculum progresses.

5 Performance on real-world tasks

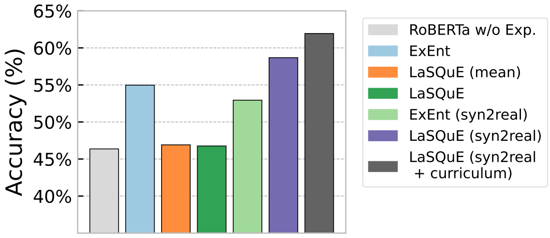


Figure 7: Zero-shot classification accuracy on novel real-world classification tasks in CLUES-Real.

Comparison with ExEnt. In the previous sections, we established the effectiveness of our proposed strategies on a large number of synthetic tasks from CLUES. Here, we empirically evaluate LaSQuE on the 36 real-world classification tasks from CLUES using the aforementioned strategies.

In Figure 7, we find that directly trying to train LaSQuE fails to surpass the baselines (even when using attention to aggregate over explanations) as the comparatively low number of explanations in CLUES-Real hinders the model from learning quantifier semantics and classification jointly. To alleviate this issue, we pre-train on the synthetic tasks and then fine-tune the learned model on the real tasks, which we also see as a natural type of curriculum learning. We find that pre-training on synthetic tasks (LaSQuE (syn2real)) gives a relative gain of 6.7% in generalization accuracy over ExEnt. On the contrary, if we pre-train ExEnt using the same set of synthetic tasks, we find that the resultant model, ExEnt (syn2real), is inferior to ExEnt in terms of generalization accuracy (as shown in figure 7). LaSQuE (syn2real) outperforms ExEnt (syn2real) (58.68% vs 52.94%), showing that LaSQuE is better in transferring the skills learned over synthetic tasks to real-world tasks. Next, we evaluate the utility of curriculum learning on real tasks. We start with a pre-trained LaSQuE on synthetic tasks and then fine-tune it first on binary tasks of CLUES-Real followed by training on multiclass tasks of CLUES-Real. We find that curriculum learning results in the best generalization (LaSQuE (syn2real + curriculum)) performing significantly better than ExEnt (relative gain of 12.7%; $p < 0.005$, paired t-test) on CLUES-Real.

Comparison with Large Instruction-tuned models. Recent works show that large language models (LLMs) fine-tuned on multiple classification tasks have an ability for zero-shot classification on new tasks (Ouyang et al., 2022; Sanh et al., 2022; Chung et al., 2022). These models have been primarily trained on unstructured text classification tasks and instructions that define the task rather than providing logic for classification. Given the emergent ability of large language models (Wei et al., 2022), we test the performance of such models on CLUES-Real and compare them with our best LaSQuE model. Specifically, we compare against the publicly available T0-3B (Sanh et al., 2022) and FLAN-T5-XXL (Chung et al., 2022) models. We report the generalization accuracy over 16 real-world tasks with and without using explanations in the prompt for T0-3B and FLAN-T5-XXL in Table 2. We find that both T0-3B and FLAN-T5-XXL with explanations in the prompt (47.90% and 42.30% respectively) perform worse than our best LaSQuE (61.90%) on the same set of tasks. This shows that our strategies for LaSQuE instill stronger inductive biases into a much smaller model (125M for LaSQuE vs 3B for T0/ 11B for FLAN-T5-XXL). Further, adding explanations in the prompt lowers performance of both T0-3B and FLAN-T5-XXL showing that these models struggle in understanding the classification logic described in form of natural language explanations, for structured classification tasks. Future work should explore improved techniques for using large instruction-tuned models under zero-shot settings for structured classification tasks guided by natural language explanations.

METHOD	ACCURACY (↑)
LaSQuE (best)	**61.90%**
T0-3B (w/o exp.)	49.47%
T0-3B (w/ exp.)	47.90%
FLAN-T5-XXL (w/o exp.)	44.47%
FLAN-T5-XXL (w/ exp.)	42.30%

Table 2: Zero-shot accuracy of our best LaSQuE model, T0-3B, and FLAN-T5-XXL (11B) on 16 unseen classification tasks of CLUES-Real.

6 Conclusion

We have presented effective and generalizable strategies to learn classifiers from language explanations. While our results are promising, our analysis also highlights several open challenges in learning from language. In particular, LaSQuE struggles to learn quantifier semantics without quantifier-specific supervision (in the form of pre-defined initialization or ordinal relations), especially when tasks have complex explanations (due to the presence of negations/conjunctions/disjunctions). Further, our modeling of quantifiers as fixed probability values is restrictive. Future work can also explore explicit modeling of negations, conjunctions and disjunctions for learning from explanations.

7 Limitations

In this work, we introduce LaSQuE, which models and learns the differential semantics of linguistic quantifiers present in natural language explanation to train a classifier guided by these explanations. We evaluate the efficacy of LaSQuE over baselines on the CLUES benchmark.

This work assumes that only a single quantifier is present in the explanations. However, in real-world settings, explanations may contain multiple quantifiers. Modeling the composition of quantifiers can be an interesting direction for future work to make the paradigm of learning from explanations more robust toward fuzzy concepts expressed in real-world explanations.

For our experiments, we assume perfect extraction of quantifiers and limit our analysis to a limited set of quantifiers in this work. Furthermore, we assume that the effect of quantifiers in a sentence is the same irrespective of the domain of the sentence. For example, consider two sentences *'pungent mushrooms are usually toxic'* and *'people who smoke regularly usually suffer from cancer'*. Here the effect of *'usually'* is not exactly the same for two sentences that are from different domains. However, LaSQuE is not sensitive to the task domain while modeling the semantics of the quantifier. Future work can investigate variations in the semantics of the same quantifier across different domains and also how to incorporate/learn such domain-specific differences (for example, by modeling the semantics of a quantifier as a probability distribution rather than a point value).

Ethics and Broader Impact

All our experiments are performed over publicly available datasets, specifically datasets (including language explanations) from CLUES benchmark (Menon et al., 2022). The datasets do not contain any information that uniquely identifies the crowdworkers involved in data collection. We do not perform any additional annotation or human evaluation in this work.

Our method, LaSQuE can learn classifiers over structured data using language explanations provided as part of input to the classifier. LaSQuE is built over existing pre-trained language model, RoBERTa (Liu et al., 2019). We do not foresee any risks with our method if the inputs to our model are appropriate for the task. Any measures to counteract erroneous inputs (that may be provided deliberately, potentially exploiting unwanted biases) or curb the biases of pre-trained language models are beyond the scope of this work.

The broader impact of this research in the longer term could increase the accessibility of predictive technologies for ordinary users (non-experts), enabling them to customize AI technologies through natural language interactions.

Acknowledgments

This work was supported in part by NSF grant DRL2112635. The views contained in this article are those of the authors and do not necessarily reflect the views or opinions of the funding agency.

References

Jacob Andreas, Dan Klein, and Sergey Levine. 2018. Learning with latent language. 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 2166–2179, New Orleans, Louisiana. Association for Computational Linguistics.

Forough Arabshahi, Kathryn Mazaitis, Toby Jia-Jun Li, Brad A Myers, and Tom Mitchell. 2020. Conversational learning.

Elke Bach, Eloise Jelinek, Angelika Kratzer, and Barbara BH Partee. 2013. *Quantification in natural languages*, volume 54. Springer Science & Business Media.

Jon Barwise and Robin Cooper. 1981. Generalized quantifiers and natural language. In *Philosophy, language, and artificial intelligence*, pages 241–301. Springer.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 41–48.

Duo Chai, Wei Wu, Qinghong Han, Wu Fei, and Jiwei Li. 2020. Description based text classification with reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Sahil Chopra, Michael Henry Tessler, and Noah D. Goodman. 2019. The first crank of the cultural ratchet: Learning and transmitting concepts through language. In *CogSci*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Ruixiang Cui, Daniel Hershcovich, and Anders Søgaard. 2022. Generalized quantifiers as a source of error in multilingual NLU benchmarks. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4875–4893, Seattle, United States. Association for Computational Linguistics.

Avia Efrat and Omer Levy. 2020. The turking test: Can language models understand instructions? *arXiv preprint arXiv:2010.11982*.

Sheng Guo, Weilin Huang, Haozhi Zhang, Chenfan Zhuang, Dengke Dong, Matthew R Scott, and Dinglong Huang. 2018. Curriculumnet: Weakly supervised learning from large-scale web images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 135–150.

Guy Hacohen and Daphna Weinshall. 2019. On the power of curriculum learning in training deep networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2535–2544. PMLR.

Braden Hancock, Paroma Varma, Stephanie Wang, Martin Bringmann, Percy Liang, and Christopher Ré. 2018. Training classifiers with natural language explanations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1884–1895, Melbourne, Australia. Association for Computational Linguistics.

Austin W Hanjie, Ameet Deshpande, and Karthik Narasimhan. 2022. Semantic supervision: Enabling generalization over output spaces. *arXiv preprint arXiv:2202.13100*.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature*, 585(7825):357–362.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR.

Eric Jones, Travis Oliphant, Pearu Peterson, et al. 2001–. SciPy: Open source scientific tools for Python.

Howard S Kurtzman and Maryellen C MacDonald. 1993. Resolution of quantifier scope ambiguities. *Cognition*, 48(3):243–279.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sebastian Lobner. 1986. Quantification as a major module of natural language semantics. In *Studies in discourse representation theory and the theory of generalized quantifiers*, pages 53–86. De Gruyter.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. 2019. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *International Conference on Learning Representations*.

Lingjie Mei, Jiayuan Mao, Ziqi Wang, Chuang Gan, and Joshua B. Tenenbaum. 2022. FALCON: Fast visual concept learning by integrating images, linguistic descriptions, and conceptual relations. In *International Conference on Learning Representations*.

Rakesh R Menon, Sayan Ghosh, and Shashank Srivastava. 2022. CLUES: A benchmark for learning classifiers using natural language explanations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6523–6546, Dublin, Ireland. Association for Computational Linguistics.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (to appear)*.

Linda M. Moxey and Anthony J. Sanford. 1986. Quantifiers and Focus. *Journal of Semantics*, 5(3):189–206.

Linda M Moxey and Anthony J Sanford. 1993. Prior expectation and the interpretation of natural language quantifiers. *European Journal of Cognitive Psychology*, 5(1):73–91.

Sanmit Narvekar, Jivko Sinapov, and Peter Stone. 2017. Autonomous task sequencing for customized curriculum design in reinforcement learning. In *IJCAI*, pages 2536–2542.

Elissa L Newport. 1990. Maturational constraints on language learning. *Cognitive science*, 14(1):11–28.

Rasha Obeidat, Xiaoli Fern, Hamed Shahbazi, and Prasad Tadepalli. 2019. Description-based zero-shot fine-grained entity typing. 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 807–814, Minneapolis, Minnesota. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Georgios Pavlakos, Xiaowei Zhou, and Kostas Daniilidis. 2018. Ordinal depth supervision for 3d human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7307–7316.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. 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 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Stephanie Solt. 2009. The semantics of adjectives of quantity.

Shashank Srivastava, Igor Labutov, and Tom Mitchell. 2017. Joint concept learning and semantic parsing from natural language explanations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1527–1536, Copenhagen, Denmark. Association for Computational Linguistics.

Shashank Srivastava, Igor Labutov, and Tom Mitchell. 2018. Zero-shot learning of classifiers from natural language quantification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 306–316, Melbourne, Australia. Association for Computational Linguistics.

Shane Steinert-Threlkeld. 2021. Quantifiers in natural language: Efficient communication and degrees of semantic universals. *Entropy*, 23(10):1335.

Yi Tay, Shuohang Wang, Anh Tuan Luu, Jie Fu, Minh C. Phan, Xingdi Yuan, Jinfeng Rao, Siu Cheung Hui, and Aston Zhang. 2019. Simple and effective curriculum pointer-generator networks for reading comprehension over long narratives. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4922–4931, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Orion Weller, Nicholas Lourie, Matt Gardner, and Matthew E. Peters. 2020. Learning from task descriptions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1361–1375, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. 2020. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, Online. Association for Computational Linguistics.

Ilker Yildirim, Judith Degen, Michael Tanenhaus, and Florian Jaeger. 2013. Linguistic variability and adaptation in quantifier meanings. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 35.

A Details on CLUES

CLUES (Menon et al., 2022) is a recently proposed benchmark of classification tasks paired with natural language explanations. The benchmark consists of 36 real-world classification tasks (CLUES-Real) as well 144 synthetic classification tasks (CLUES-Synthetic). The tasks and explanations of the benchmark are in English language. The real-world classification tasks were created using resources from UCI Machine Learning repository, Kaggle, and Wikipedia tables. The explanations for real-world tasks were crowdsourced. The synthetic tasks were created programmatically to study the performance of models under different levels of task complexities. The 48 different complexities in CLUES-Synthetic arise from the: (a) presence of negations in clauses and/or labels, (b) structure of explanations (conjunctions/disjunctions/nested), (c) presence of quantifiers in explanations, and (d) binary vs multiclass classification task. The explanations for CLUES-Synthetic are generated programmatically using templates. In this work, we follow the train and test splits for CLUES-Real from Menon et al. (2022). Additionally, we train on 70% of the labeled examples of the seen tasks and perform zero-shot generalization test over the 20% examples of each task in CLUES-Real. For the extremely small tasks, we use the entire set of examples for zero-shot testing. The seen-unseen task splits for CLUES-Real and CLUES-Synthetic that we use for experiments in this paper is the same as that in Menon et al. (2022).

A.1 List of quantifiers

The full list of quantifiers along with their associated probability values are shown in Table 3.

QUANTIFIERS	PROBABILITY
"always", "certainly", "definitely"	0.95
"usually", "normally", "generally", "likely", "typically"	0.70
"often"	0.50
"sometimes", "frequently",	0.30
"occasionally"	0.20
"rarely", "seldom"	0.10
"never"	0.05

Table 3: Probability values used for quantifiers in CLUES. These values are based on Srivastava et al. (2018).

B Training details

In this section we provide details about implementation such as hyperparameter details, and details about hardware and software used along with an estimate of time taken to train the models.

B.1 Hyper-parameter settings

For all the transformer-based models we use the implementation of HuggingFace library (Wolf et al., 2020). All the model based hyper-parameters are thus kept default to the settings in the HuggingFace library. We use the publicly available checkpoints to initialize the pre-trained models. For RoBERTa based baselines we use 'roberta-base' checkpoint available on HuggingFace. For our intermediate entailment model in ExEnt, we fine-tune a pre-trained checkpoint of RoBERTa trained on MNLI corpus ('textattack/roberta-base-MNLI' from HuggingFace).

When training on CLUES-Synthetic, we use a maximum of 64 tokens for our baseline RoBERTa w/o Exp. and ExEnt.

We used the AdamW (Loshchilov and Hutter, 2019) optimizer commonly used to fine-tune pre-trained Masked Language Models (MLM) models. For fine-tuning the pre-trained models on our benchmark tasks, we experimented with a learning rate of $1e-5$. In order to learn the quantifier probabilities, we search for the correct learning rate to use in $\{1e-3, 2e-3, 5e-3, 9e-3, 1e-2, 2e-2, 3e-2\}$ and use $1e-2$ for our reported experiments based on the best validation accuracy obtained while training and testing on the binary classification datasets with no negation and conjunction complexities in explanations/concepts. Batch sizes was kept as 2 with gradient accumulation factor of 8. The random seed for all experiments was 42. We train all the models for 20 epochs. Each epoch comprises of 100 batches, and in each batch the models look at one of the tasks (in a sequential order) in the seen split. In the curriculum learning experiments, we run the model on each task type for 20 epochs and select the best model during a particular step of the curriculum based on the validation scores of the seen tasks. Finally, the chosen best checkpoint is used to initialize the model for the next step of the curriculum.

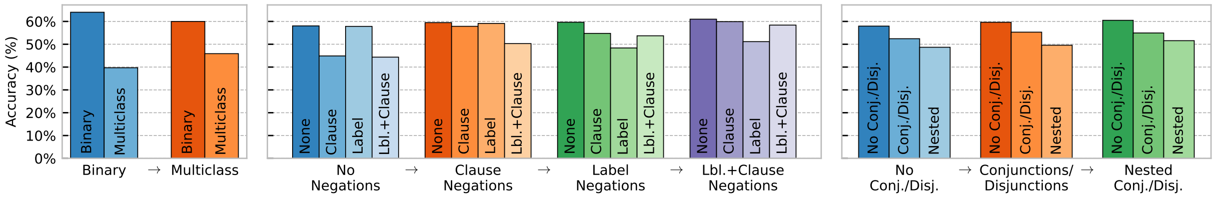


Figure 8: Progression of generalization accuracies on task complexities as we move forward in the curriculum for all three curricula (from left to right: Classes, Negations and Conjunctions curriculum). The text on each bar indicates the evaluation complexity, while the x-axis indicates the complexity that the model has been currently trained on in the curriculum.

B.2 Hardware and software specifications

All the models are coded using Pytorch 1.4.0[5] (Paszke et al., 2019) and related libraries like numpy (Harris et al., 2020), scipy (Jones et al., 2001–) etc. We run all experiments on a Tesla V100-SXM2 GPU of size 16GB, 250 GB RAM and 40 CPU cores.

B.3 Training times

- Training on CLUES-Real: The baseline RoBERTa w/o Exp model typically takes 3 seconds on average for training on 1 batch of examples. ExEnt and LaSQuE (all variants) also take comparable amount of time to train on 1 batch. In 1 batch, the models go through 16 examples from the tasks in seen split.
- Training on CLUES-Synthetic: All the models take comparatively much lesser time for training on our synthetic tasks owing to lesser number of explanations on average for a task. For training on 1 batch, all models took 1 seconds or less to train on 1 batch of examples from CLUES-Synthetic.
- Training for curriculum learning: The run time of a curriculum learning episode depends on the number of tasks in an episode. In Figure 6, the binary-multiclass curriculum takes 2 hours to train, while negations take 4 hours, and conjunctions take 3 hours. The same time frame applies for the results in Figure 8.

C Extended Analysis of Curriculum Learning

In Figure 8, we show the trajectories of generalization performance as we increase the complexity along three independent axes in the three curricula. Briefly, our results indicate that in learning tasks with more classes, generalization increases on multiclass classification tasks at the expense of a slight performance decrease on the more straightforward binary tasks. In the curriculum focused on negations, LaSQuE underperforms on tasks with explanations that have 'label negations' after training on the relevant training datasets for that complexity. However, on further analysis, we observe that this trend is more pronounced when 'label negations' are paired with multiclass classification tasks. By contrast, LaSQuE improves through training on the relevant training datasets of binary classification tasks with 'label negations' in concepts. Lastly, training progressively on more structurally complex tasks resulting from conjunctions/disjunctions in explanations shows improvements during evaluation across all conjunction types without forgetting how to solve simpler tasks.

[5] https://pytorch.org/

Model Name	No Conjunctions No Quantifiers				Simple Conjunctions No Quantifiers				Nested Conjunctions No Quantifiers			
	NN	CN	LN	BN	NN	CN	LN	BN	NN	CN	LN	BN
LaSQuE (predefined init)	89.67	87.09	**81.40**	42.87	**78.05**	**68.41**	51.01	56.84	75.38	65.82	**58.70**	45.14
LaSQuE (scratch)	89.67	87.09	**81.40**	42.87	**78.05**	**68.41**	51.01	56.84	75.38	65.82	**58.70**	45.14
LaSQuE (ordinal)	**91.22**	**88.00**	79.79	**59.97**	76.00	62.91	**54.05**	**58.02**	74.72	**68.66**	51.62	**47.24**
ExEnt	89.67	87.09	**81.40**	42.87	**78.05**	**68.41**	51.01	56.84	75.38	65.82	**58.70**	45.14

Model Name	No Conjunctions With Quantifiers				Simple Conjunctions With Quantifiers				Nested Conjunctions With Quantifiers			
	NN	CN	LN	BN	NN	CN	LN	BN	NN	CN	LN	BN
LaSQuE (predefined init)	64.81	**56.77**	**63.70**	56.16	56.30	**55.02**	49.79	44.37	**54.77**	46.80	44.66	40.79
LaSQuE (scratch)	**66.15**	47.48	47.01	47.33	54.00	44.76	49.90	42.20	52.23	35.46	42.70	37.15
LaSQuE (ordinal)	65.10	45.03	56.88	**56.83**	**56.36**	50.41	**53.08**	**51.78**	53.48	**49.22**	**44.92**	**43.91**
ExEnt	49.41	40.02	47.88	43.44	49.17	42.87	45.84	42.33	43.34	37.67	37.41	34.80

Table 4: Classification accuracies of LaSQuE and ExEnt. 'NN', 'CN', 'LN', and 'BN' stand for 'no negations', 'clause negations', 'label negations', and 'both negations', respectively denoting the variations of negations appearing in the explanations of CLUES-Synthetic. We see that LaSQuE outperforms ExEnt (Menon et al., 2022) on all tasks having explanations with quantifiers. Within each set, task complexity increases from left to right due to negations.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 7

☑ A2. Did you discuss any potential risks of your work?
Section 8

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

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

B ☑ Did you use or create scientific artifacts?

Section 3.1

☑ B1. Did you cite the creators of artifacts you used?
Introduction, Section 3.1

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
We do not create any new datasets besides what already exists in prior work. The CLUES benchmark we utilize does not have an associated license. Our main baseline, \exent, is available under the MIT License.

☑ 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)?
Section 3.1

☒ 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?
The dataset that has been used in this work describes "how to solve a classification task?". Hence, the kind of textual data used in our work depends on the classification task provided by prior work in some cases. Further, the paper accompanying the dataset mentions that the dataset does not contain any personal information about the crowdworkers involved during its creation.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
We report language of datasets used in Appendix A.
Note: We exclusively use already available datasets for our experiments in this paper. In the CLUES benchmark, quantifiers are present in 50% explanations as per the benchmark paper. Hence, we did not report any additional analysis on the number. Besides quantifiers, we do not use any other linguistic phenomena or demographic information to improve our classifiers.

☑ 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.
Appendix A

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

C ☑ Did you run computational experiments?

Section 3

☑ 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 B

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

☑ 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?
Section 4 and Appendix B

☑ 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.)?
We use zero-shot generalization accuracy as a metric. This quantity was computed manually. We use HuggingFace Transformers and Pytorch in our code and cite them appropriately. More in Appendix B

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.