When More Data Hurts: A Troubling Quirk in Developing Broad-Coverage Natural Language Understanding Systems

Anonymous ACL submission

Abstract

In natural language understanding (NLU) production systems, the end users' evolving needs necessitate the addition of new abilities, indexed by discrete symbols, requiring additional training data and resulting in dynamic, ever-growing datasets. Dataset growth introduces new challenges: we find that when learning to map inputs to a new symbol from a fixed number of annotations, more data can in fact *reduce* the model's performance on examples that involve this new symbol. We show that this trend holds for multiple models on two datasets for common NLU tasks: intent recognition and semantic parsing (see Fig. 1). We demonstrate that the performance decrease is largely associated with an effect we refer to as source signal dilution (cf. Fig. 2), which occurs when strong lexical cues in the training data become diluted as the dataset grows. Selectively dropping training examples to prevent source signal dilution often reverses the performance decrease, suggesting a promising direction for addressing this issue.[1]

1 Introduction

Broad-coverage natural language understanding (NLU) that simultaneously supports a wide range of user requests is critical for developing general-purpose natural language interfaces. Such systems are currently being deployed and reach millions of users worldwide: As of 2021, Amazon Alexa contains more than 80,000 different skills (Vailshery, 2021), and Microsoft has deployed a new conversational interface for Outlook that uses over 300 composable functions to represent fine-grained semantics in task-oriented user-agent dialogues (Semantic Machines et al., 2020; Burrage, 2021).

These broad-coverage NLU systems do not acquire their full capability on day one: new features (e.g., intents or functions) are incrementally added over time, along with new supervised training data

[1]Our code and data are available at `anonymous-link`.

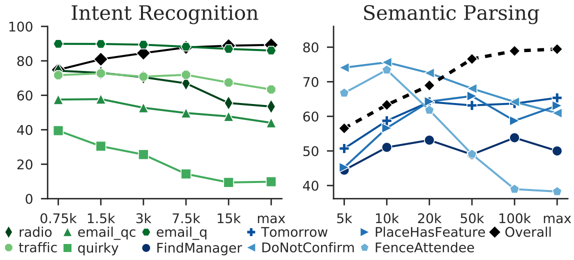

Figure 1: Overall and per-symbol test accuracy for intent recognition and semantic parsing when the number of training examples for a certain symbol is fixed. Each line represents fixing a different symbol. As training data size increases, overall accuracy increases but accuracy for the fixed symbol often decreases.

for learning the new features. However, there has been little research on the data and learning dynamics during such incremental development, which is critical considering the wide deployment and high cost of such systems. This work aims to bring attention to this important problem. We consider two prototypical NLU tasks: intent recognition and semantic parsing. NLU generally consists of mapping utterances into a space of *symbols* or symbol sequences (e.g., intent labels for intent recognition and sequences of functions/predicates for semantic parsing; we refer to both cases as "symbols" for ease of discussion). We consider the following incremental development process (which is typical in practice): given a set of existing symbols and their training data, we want to learn a new symbol, which entails adding new annotations for the symbol. As the system supports more and more symbols, its training data size continually increases.

At first blush, this growth may seem positive, in holding with a common assumption of supervised learning: *more data is generally better* (Kearns et al., 1994). However, our analyses reveal a troubling quirk: *as the training data size increases, it becomes increasingly difficult to learn new symbols.* To investigate this further, we create datasets of increasing size by adding a fixed number of training

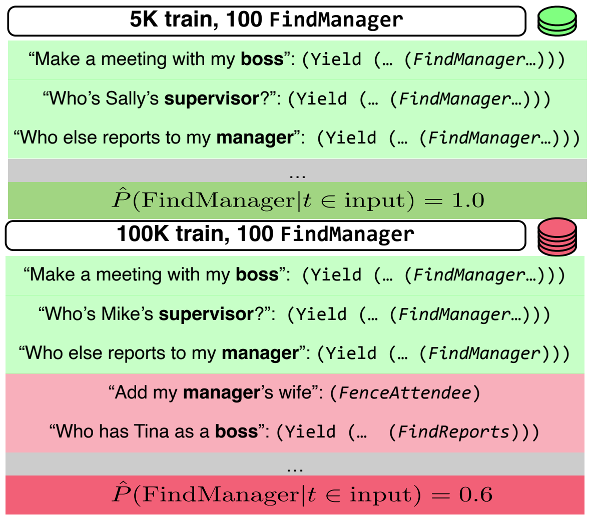

Figure 2: Source signal dilution in the training set: as the data grows, the set of cues t (in bold) associated with `FindManager` becomes less predictive of it.

examples for a new symbol into increasingly larger datasets of examples for other existing symbols, simulating learning a new symbol with larger and larger datasets. We then train on each setting and evaluate on the same test set, measuring test accuracy on the new symbol. We repeat this across an intent recognition dataset (Liu et al., 2019) and a semantic parsing dataset (Semantic Machines et al., 2020), examining 5 symbols per dataset.

Fig. 1 shows the overall test accuracy of our best models (cf. §2) as well as the accuracy on examples containing the new symbol. As the size of the dataset increases, the average test accuracy across all symbols monotonically increases. However, the accuracy on the examples for the new symbol generally decreases. The decrease in performance could lead to a *vicious cycle*, whereby an increasing number of training examples would need to be collected to achieve adequate accuracy for each new symbol, which accelerates the growth of the dataset and in turn increases the demand for training data for future symbols, and possibly also for existing symbols, as they become a smaller percentage of the data, even further. Class imbalance is one obvious candidate explanation for the performance decrease: as the dataset grows and the number of examples for the new symbol stays fixed, the prior probability of the new symbol in the training data decreases. If this were true, simply upsampling the new symbol's annotations should revert the decrease. With a view to addressing this kind of class imbalance, we explore two common solutions: group distributionally robust optimization (DRO) (Sagawa et al., 2019, 2020) and upsampling. The failure of these solutions to attenuate the accuracy drop leads us to identify a different force associated with the performance decrease, *source signal dilution*, whereby the reliability of the signal coming from indicative tokens in the user utterances for the new symbol is diminished in larger datasets. This force is illustrated in Fig. 2. At low data settings, some tokens are highly correlated with the `FindManager` symbol, but as the dataset grows, the correlation with these tokens is reduced by competing examples that, often by coincidence, contain the same tokens. We show that when confounding examples (shown in red) are removed, the accuracy decrease largely disappears, indicating that our state-of-the-art neural models are overly-reliant on simple lexical cues for learning the symbol of interest. We later argue that while the removal of these examples may cure the symptoms expressed under increasing dataset sizes, they do not represent an adequate cure.

Our main contributions are threefold:

- We identify a troubling quirk in developing broad-coverage NLU systems that challenges the common assumption of more data entailing better performance.
- Based on our observations, we identify plausible forces leading to the decreased accuracy seen in Fig. 1, foremost among them the dilution of the source signal (cf. Fig. 2). A deeper understanding of this force may guide us in developing systematic solutions to this problem in the future.
- Finally, we release our code and models, including a model for the SMCalFlow dataset (Semantic Machines et al., 2020) that achieves state-of-the-art performance. We hope it will serve as a useful baseline for future development on this challenging dataset.

2 Datasets and Models

Intent recognition and task-oriented semantic parsing share multiple common features. They both translate user utterances to structured objects (or at least to categorical intents) and they are both commonly used in production NLU systems trained on annotated examples. This makes them ideal testbeds for exploring the dynamics in learning new symbols as the training dataset grows.

2.1 Intent Recognition

Intent recognition involves classifying utterances into a fixed set of “intents,” which are typically the symbols of some agent (e.g., a digital personal assistant) (Lorenc et al., 2021). Intents often index into a set of pre-defined templates (e.g., the intent `play_music` might index into a template with slots “song name,” “song artist” etc.) and are central to many digital assistant technologies. New intents may be added to the agent incrementally during the development process as needs for new capabilities arise. For example, an agent capable of cooking tasks may be extended to other household tasks, requiring it to understand the associated intents.

We use the *NLU evaluation* dataset provided by Liu et al. (2019), which contains 25,715 utterances for 68 intents across 18 scenarios. 2571 and 5144 utterances are reserved for validation and testing, respectively. We simulate learning a new intent under different data regimes by choosing an intent to learn, and then sampling examples for that intent and the rest of the dataset at different ratios. The number of examples for the new intent is fixed at 30, and we vary the size of the dataset $N \in \{750, 1500, 3000, 7500, 15{,}000, 18{,}000\}$ where 18,000 is the *max* in Fig. 1. Our experiments span 5 intents:

- `play_radio` is primarily triggered when users ask for radio stations to be played.
- `email_query` is for email-related queries.
- `email_querycontact` is triggered by questions about contacts in an address book.
- `general_quirky` is a catch-all category for trivia-style questions and pleasantries.
- `transport_traffic` is triggered by traffic-related questions and commands.

Some of the intents (e.g., `play_radio`) have a set of easily-identified input triggers (e.g., “radio”, “fm”) while others (e.g., `general_quirky`) have very diverse inputs. Example utterances for each intent can be found in Appendix A.1.

To model this data, we apply a linear classification layer to the `[CLS]` token of BERT base (Devlin et al., 2019), finetuning the whole contextualized encoder at training time. This model was trained to convergence with the Adam optimizer, using a learning rate of 1e−5.

2.2 Semantic Parsing

While providing a good environment for experimentation, the intent recognition task lacks the complexity of a full real-world production environment. We therefore seek to expand our intent recognition experiments and analyses to a production-level task and dataset. To that end, we use the SMCalFlow dataset (Semantic Machines et al., 2020; Burrage, 2021), which offers a task-oriented semantic parsing challenge, where a user iteratively creates a dataflow graph in a dialogue with an agent (cf. Fig. 3). The dataset has 41,517 dialogues with 338 function types, yielding 121,024 training user turns in the full setting. The input to our parsing model is the previous user utterance, the corresponding agent response, and the current user utterance, all concatenated. The model is tasked with learning to generate a typed Lisp program; see Fig. 3 for an example, with further examples in Appendix A.2.

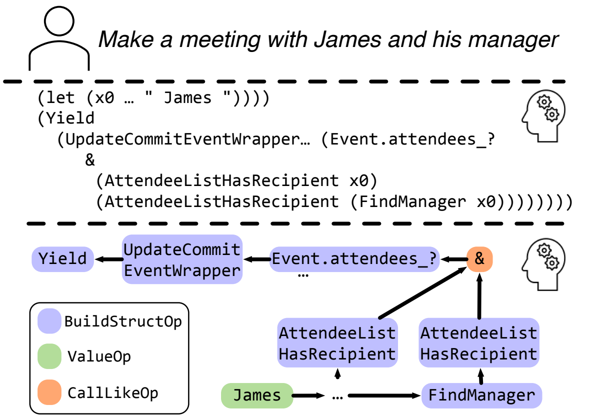

Figure 3: Example SMCalFlow program; it can be represented as a Lisp expression or as a DAG.

We explore both a sequence-to-sequence (seq2seq) model and a sequence-to-graph (seq2graph) model, using the MISO framework (Zhang et al., 2019b; Stengel-Eskin et al., 2021), which is built on top of AllenNLP (Gardner et al., 2018). The former directly predicts the Lisp string, while the latter produces a DAG as seen at the bottom of Fig. 3. Details follow.

LSTM seq2seq Our baseline model is an LSTM-based seq2seq model similar to that used by Dong and Lapata (2016) and Semantic Machines et al. (2020). The model consists of a BiLSTM encoder and an LSTM decoder with attention over the encoder states and a source copy operation to copy entity spans from the source text, with its embedding layer initialized from GloVe embeddings (Pennington et al., 2014). Additional model details are given in Appendix B.1.

Transformer-based transductive A more competitive approach follows the transductive parsing paradigm (Zhang et al., 2019a) which aims to directly produce the underlying DAG instead of the surface form, generating graph nodes as well as edges. We implement a transformer-based transductive model, based on the architecture and code from Stengel-Eskin et al. (2021). The model directly generates the linearized DAG (cf. Fig. 3) underlying the SMCalFlow Lisp expression; the nodes of the DAG (functions and arguments) are generated in a sequence-to-sequence fashion, with graph edges and edge types being assigned during decoding via a biaffine parser (Dozat and Manning, 2017). Following past work, the input features for this model are a concatenation of BERT (Devlin et al., 2019), GloVe, and character CNN features. Additional details on the transductive transformer model are given in Appendix B.2.

Data As SMCalFlow's test set is not publicly available, we first split the validation data in half to obtain a held-out test set. We then construct training splits by selecting 100 examples for each symbol and varying $N \in \{5000, 10{,}000, 20{,}000, 50{,}000, 100{,}000, \mathit{max}\}$, where *max* is the maximum amount of data that can be taken from the 121,024 training examples while excluding all but 100 examples for the new symbol. This number is typically slightly under 120,000. We describe the symbols examined; Appendix A.2 has further examples.

- `FindManager` is invoked when a user queries for a person's manager.
- `Tomorrow` returns tomorrow's date.
- `DoNotConfirm` is applied when a user wants to cancel a proposed action (e.g., confirming the creation of an event) by the agent.
- `FenceAttendee` is invoked when a user tries to make an event with an attendee who is not in their contact list.
- `PlaceHasFeature` is used when a user asks whether a place has certain amenities (e.g., outdoor seating) or offers certain services.

These vary in compositionality: `FindManager` and `PlaceHasFeature` take arguments, but `Tomorrow` does not, and `FenceAttendee` and `DoNotConfirm` are complete programs in themselves. `FindManager` and `Tomorrow` have strong input cues, but `DoNotConfirm` and `FenceAttendee` come from very diverse inputs.

3 Experiments

Baseline The first experimental setting is the baseline setting, as presented in Fig. 1. Here, we vary the size of the training data while holding the number of examples for each given symbol fixed. We train each model with three random seeds for all experiments, reporting the average.

Upsampling Given the the limited number of examples for the new symbol, upsampling is a natural solution to attempt. After exploring upsampling ratios $r \in \{2, 4, 8, 16, 32, 64\}$, we selected 32 as the factor by which to upsample the new symbol examples based on validation performance, i.e. each example for the new symbol is copied 32 times in the training set. This setting addresses both the class imbalance (the class corresponding to the new symbol is now 32 times more likely) and to some extent the decreased reliability of source triggers at higher dataset sizes (i.e., source signal dilution), since it upsamples examples with a reliable trigger-to-symbol mapping.

Group DRO has been proposed as a method for robust generalization under severe class imbalances (Sagawa et al., 2019). Rather than optimizing by minimizing the average loss across a training batch, group DRO seeks to minimize the loss for the worst-performing group in each batch. More formally, given a set of groups $\mathcal{G}$, a parameter space Θ, a network $f(x;\theta)$, a loss l, and a per-group training distribution P_g, the group DRO objective is:

$$\theta^* = \arg\min_{\theta\in\Theta} \left(\max_{g\in\mathcal{G}} \mathbb{E}_{(x,y)\sim P_g} \left[l(f(x;\theta), y) \right] \right)$$

We apply this objective to our intent recognition model, treating each intent as a separate group. As long as the worst-performing group is the intent of interest (e.g., `play_radio`), the model will be optimized solely for that intent. For the SMCalFlow setting, applying group DRO is more challenging, as the output is a program containing multiple functions rather than a single class. We apply group DRO to SMCalFlow by defining two groups: programs with the new symbol, and those without it.

4 Results and Analysis

4.1 Overall Model Performance

For intent recognition, the BERT-based classifier, when trained on the full dataset, obtains 90.49% test accuracy, indicating that it is suited to the task.

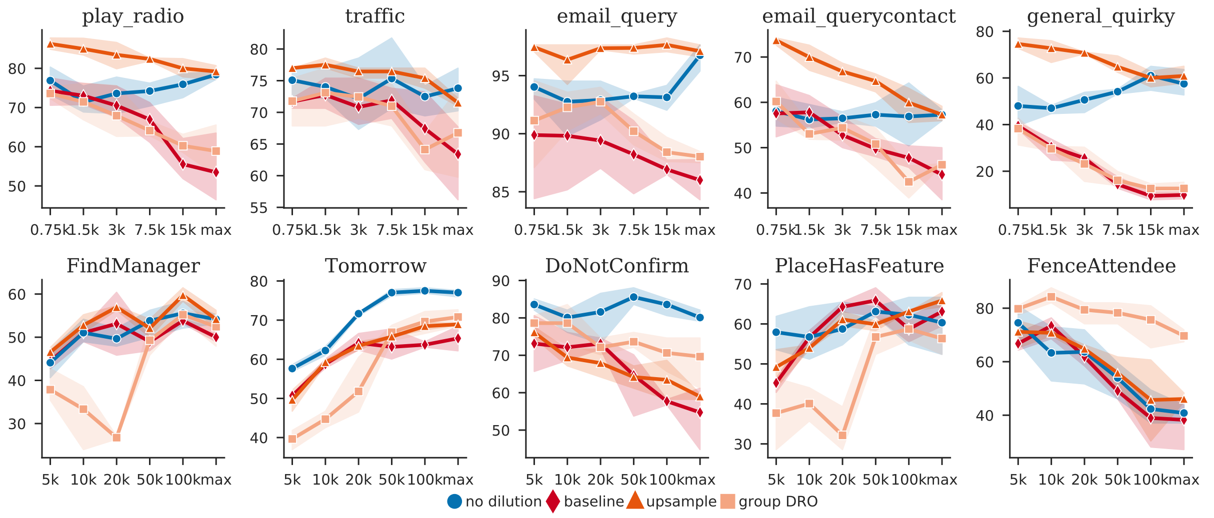

Figure 4: Per-symbol accuracy on intent recognition and semantic parsing as the size of the training set increases. Shaded regions represent 95% bootstrap confidence intervals.

Table 1 shows the performance of the semantic parsing models on the full validation and test splits of SMCalFlow. Here, the test split is the held-out split used in the official SMCalFlow leaderboard. To further increase the Transformer model's performance, we follow Stengel-Eskin et al. (2020) and unfreeze the top 8 layers of BERT. This model outperforms Platanios et al. (2021), and represents the current state-of-the-art SMCalFlow parser. For the following experiments, we use the model without fine-tuning BERT in order to reduce computation.

Model	**Validation EM**	**Test EM**
LSTM (ours)	66.9%	52.4%
Transformer (ours)	79.3%	74.5%
Platanios et al. (2021)	–	75.3%
Transformer tuned (ours)	80.3%	**75.5%**

Table 1: Semantic parsing exact-match (EM) performance when trained on the full dataset. The transductive model with encoder tuning is the state of the art.

4.2 More Data Can Hurt Performance

Intent Recognition Fig. 1 and Fig. 4 show the overall and per-symbol accuracy of a model when the number of examples for a new intent is fixed at 30. As the size of the training set increases, the overall accuracy of the model, averaged across all intents, improves. However, the accuracy on the intents of interest decreases.

Semantic Parsing When the number of examples for a symbol is fixed at 100, and the number of other training examples increases, Fig. 1 and Fig. 4 show that the accuracy on new symbols is highly non-monotonic. Fig. 5 shows that the non-

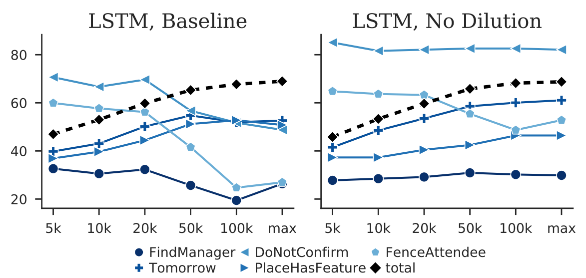

Figure 5: LSTM performance on symbols of interest decreases as the total training size increases, but removing source signal dilution largely fixes it.

monotonic accuracy is not just a quirk of transductive parsing but is also seen in a commonly-used LSTM-based seq2seq baseline model. While intent recognition displayed a largely decreasing performance curve, some curves for SMCalFlow symbols (e.g., `FindManager`) increase and decrease at different settings. This may be attributed to competing forces: additional data may increase the seq2seq or seq2graph model's fluency in producing syntactically correct outputs, but also increase the source dilution, with different settings having different balances of these forces.

4.3 Addressing Class Imbalance

As the size of the dataset grows, the ratio of other symbols to the symbol of interest grows, i.e., there is greater class imbalance, with the symbol of interest representing a smaller and smaller minority. If a growing class imbalance is the culprit behind the decrease in accuracy observed in

Fig. 1, then we should expect a robust optimization technique that prioritizes minority classes to ameliorate the problem. The group DRO curves in Fig. 4 show that, while the accuracy on the new symbol is often raised above the baseline (e.g., `FenceAttendee`, `email_query`), it generally still decreases as the training set grows. Additionally, applying group DRO often results in an accuracy lower than the baseline for many training data sizes (e.g., `PlaceHasFeature`, `FindManager`, `traffic`). Thus, while group DRO may improve the results on the new symbol for a given setting, it does not alleviate the core problem: a larger dataset leads to lower accuracy than what could have been obtained with a smaller dataset.

The upsampling curves in Fig. 4 show a similar trend: upsampling can improve overall accuracy and can reduce the rate at which the accuracy decreases, but often fails to remove the decrease. In some cases (e.g., `email_query`, `PlaceHasFeature`) upsampling does lead to monotonically-improving accuracy; however, these improvements are inconsistent across symbols, suggesting that there may be other forces at play.

4.4 Addressing Source Signal Dilution

The failure of group DRO and upsampling to solve the problem suggests that it may not be due purely to an increased class imbalance between the symbol of interest and the other symbols. An additional contributing factor might be a decrease in the reliability of the source signal as additional data is added. For many symbols, there is often a set of tokens $\mathcal{T}$ that can be found in most of the utterances for that symbol. For example, for the `play_radio` intent, at least one of the tokens in $\mathcal{T} = \{\texttt{radio}, \texttt{fm}, \texttt{play}\}$ is found in 78.04% of the corresponding utterances in the full training data. Thus, the set $\mathcal{T}$ is a strong signal for predicting `play_radio`. However, as more data is added, elements in $\mathcal{T}$ will happen to appear in the inputs for other intents, reducing their strength as a signal. In other words, the high performance on `play_radio` at lower data settings may be due to the strong signal of the elements in $\mathcal{T}$, which becomes diluted as the dataset grows.

Fig. 6 shows the empirical probability of the symbol, given that at least one of its trigger tokens appeared in the input, under the dataset. Trigger tokens were determined manually, and are given in Appendix A.3. Across symbols, as the size of the dataset increases, this probability decreases, with more examples for other symbols containing the same triggers in their inputs, diluting the signal. Note that the different starting points of these lines indicate that different symbols start with higher or lower source trigger associations. For example, even at the most concentrated setting (750 total examples to 30 `general_quirky` examples), `general_quirky` has no triggers that are strongly correlated, while `play_radio` has a set of triggers that are perfectly correlated with it. Taken together with Fig. 1, Fig. 6 shows that decreased probability of the symbol given its triggers, i.e. source signal dilution, has a positive correlation with decreased performance on that symbol.

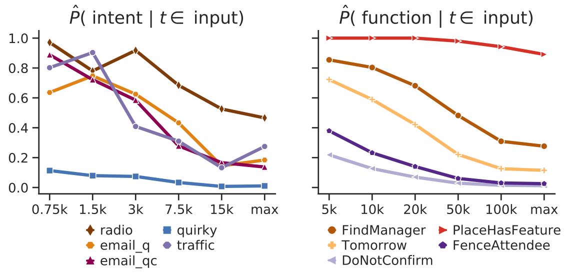

Figure 6: As the dataset size increases, the probability of the symbol given the source triggers t decreases.

Upsampling would present an intuitive solution here, as it boosts the correlations seen in Fig. 6 by increasing the number of times the symbol is seen with the input triggers, but unfortunately, as mentioned in Section 4.3, the curves in Fig. 4 indicate that upsampling does not deliver on these promises in practice.

To further investigate the impact of these diluting examples on the model performance, we experiment with removing them from the training data. In the "no dilution" setting of Fig. 4, we add the same number of examples for other symbols as in the base setting where we simply train on each data setting,[2] but we ensure that the new examples do not contain source triggers for the symbol of interest. In other words, we change the dataset such that the curves in Fig. 6 remain flat. Here, we see that the decrease is in fact often attenuated at larger training datasets, even increasing for several intents and functions, suggesting that it is in fact the reduced reliability of the source-target mapping, rather than an increase in class imbalance, that is the main factor leading to lower performance at

[2]This holds except for the max setting, where the max amount of data is reduced since more examples are discarded.

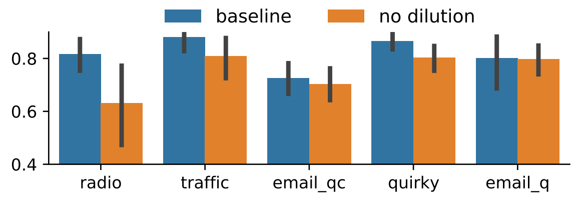

Figure 7: Test accuracy on examples that contain the triggers associated with the new intent, but whose label is some other intent.

higher data settings. This result is also not unique to BERT-based models: when we remove the same examples for the GloVe-based LSTM SMCalFlow parser, we see similar trends (cf. Fig. 5).

4.5 Impact on Competing Examples

The increase in performance on the symbol of interest that results from removing competing source examples, as seen in Fig. 4, might come at a cost. Specifically, since we are removing examples containing triggers associated with the symbol of interest (e.g., examples containing the word "manager" in the input, but without `FindManager` in the output), we run the risk of losing performance on those examples in the test set. Fig. 7 shows the intent recognition test set performance on test examples that contain triggers in the source utterance, but are not labeled with the new intent (i.e., examples that would have been excluded from the training set), averaged across training steps. The accuracy decreases when we remove diluting examples, sometimes substantially. In other words, the improvement to the new symbol from removing diluting examples comes at a cost to exactly the type of examples we are removing. While this is unsurprising, it is unfortunate, as it indicates that the strengthening of the source signal through removal, while perhaps addressing the symptom of the problem seen in Fig. 1, falls short of presenting a satisfactory cure. SMCalFlow does not have enough examples with source-side competition for other predicates in the test set to perform a similar quantitative evaluation,

5 Discussion

Firstly, our results show that as datasets grow, performance on new symbols is highly non-monotonic and often decreasing. The longer the life-span of a system, the more new symbols will be added to it – our results suggest that as a system becomes more developed, the annotation cost to the system's developer will continuously increase. Simple solutions like upsampling and group DRO do not suffice in this case: even with these in place, the performance remains non-monotonic. Our results demonstrate that removing diluting source examples does largely remove the performance decrease.

Treating this removal as a solution is, however, unsatisfying in three ways. First, we now face a new challenge as we iterate the addition of new symbols. Perhaps for the first symbol added, we can successfully remove offending examples from the training set; but as we iterate this process, we may find ourselves removing increasing percentages of our training data, with increasingly disparate subsets of annotations for each symbol. This makes removal an unattractive solution. Secondly, while we can intervene on the training distribution, we cannot control the user distribution. Fig. 7 shows that the performance on test examples containing trigger tokens but labeled with other intents does decrease after removal. This suggests that the model's ability to capture the full range of user utterances may be reduced on some axes after removal, even if the accuracy on the new symbol is increased. Finally, treating removal of diluting examples as a solution means accepting that our models are largely failing to compositionally analyze the inputs. Despite the fact that our models leverage the power of recurrence and often use large pre-trained contextualized encoders, their reliance on simple non-contextual lexical cues is reminiscent of simpler non-contextual models like Naive Bayes classifiers. We would ideally hope that a contextualized representation would be sensitive to the difference between, for example, "Who is my manager?" and "Invite my manager's wife." However, it seems that despite their contextualized inputs, the models analyzed here may be overly sensitive to the presence or absence of individual tokens.

Despite these unsatisfying observations, the removal of source-diluting examples also leads to a more promising conclusion, namely that the models we investigate seem to be able to handle large amounts of class imbalance, provided that the source signal is strong enough for the minority class. In the source removal setting, the class imbalance remains unchanged, with the ratio of examples for the new symbol to the overall training data remaining the same as for the baseline and group DRO settings. This lack of change suggests that the model can cope with large class imbalances (e.g., 100,000 total examples to 100 symbol examples) provided that the lexical cues for the minority class in the training data are strong. Both the intent classifier and the transductive parser are capable of handling extremely large class imbalance ratios if the source-target mapping is reliable. This helps explain why the model's performance can improve even as the class imbalance increases.

6 Related Work

Our learning setting relates closely to work on learning with imbalanced data as well as analyses of spurious correlations. Sagawa et al. (2020) find that over-parameterized networks display a similar trend to our trends on the worst-performing group as model capacity is increased: minority-group performance decreases as overall performance increases. They conclude that large models tend to memorize minority-class data and rely on spurious correlations, leading to worsening accuracy. Our work examines the accuracy on specific symbols as the size of the *dataset* (rather than of the model) grows. Different solutions have been proposed to improve generalization on minority data, such as distributionally robust optimization (DRO; Oren et al., 2019; Sagawa et al., 2019; Zhou et al., 2021) as well as other training and re-weighting strategies (Liu et al., 2021; Ye et al., 2021). These solutions are typically applied to image classification tasks; in a space more closely related to NLU, Li and Nenkova (2014) explore several upsampling and re-weighting strategies for discourse relation classification with imbalanced data, and Larson et al. (2019) investigate the effect of imbalanced data for detecting out-of-scope intents. Gardner et al. (2021) argue that simple lexical features, such as the ones we highlight, represent spurious correlations in the data; on this account, the models investigated here are prone to over-reliance on such spurious correlations, with the removal of diluting examples strengthening them. In a similar vein, McCoy et al. (2019) present evidence that natural language interface models rely upon spuriously-correlated features, and present a challenge dataset with such correlations mitigated.

This past work in learning with spurious correlations and imbalanced data has focused on single-label multi-class classification problems; we follow this trend in our experiments with intent recognition. However, we go beyond the single-label setting in our semantic parsing experiments, where we investigate class imbalance in a highly structured multi-label multi-class output space.

The challenging setting we present differs also from never-ending learning (Mitchell et al., 2015) and domain adaptation/continued training in that for each iteration of the dataset, a new model is trained, rather than continued training on a single model. Li et al. (2021a) investigate few-shot learning for semantic parsing via continued training, where a trained model is exposed to a small set of annotations for a new predicate. While we also attempt to learn from relatively few annotations, we do not adapt learned models, instead simulating the common production setting where models are re-trained on datasets as a whole.

Previous parsing approaches for SMCalFlow have followed both modeling paradigms used here: Semantic Machines et al. (2020) present a seq2seq baseline for SMCalFlow; this follows previous work in seq2seq semantic parsing (Vinyals et al., 2015; Dong and Lapata, 2016; Jia and Liang, 2016). Platanios et al. (2021) outperform that baseline with a transductive seq2graph model using explicit type constraints. Treating semantic parsing as a seq2graph problem has proved to be a strong paradigm for parsing Abstract Meaning Representations (Banarescu et al., 2013; Zhang et al., 2019a,b), Semantic Dependencies (Oepen et al., 2014a,b, 2016; Zhang et al., 2019b), Universal Conceptual Cognitive Annotation (Abend and Rappoport, 2013; Zhang et al., 2019b), Universal Decompositional Semantics (White et al., 2020; Stengel-Eskin et al., 2020, 2021), and GQA (Li et al., 2021b).

7 Conclusion

We examined the effect of a growing dataset on the ease of learning new symbols for NLU, finding that it often becomes harder to learn a new symbol as more data is collected. This trend holds across models and settings, and could pose significant problems as NLU systems increase in lifespan and coverage. We found that the weakening of simple lexical associations as the datasets grow is closely tied to the decrease in performance, indicating that the neural models tested in this study may be overly reliant on simple lexical cues. We end by encouraging others to examine these effects in the problems tested here and also in similar problems, where similar effects are likely to be found.

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.

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.

Eugenie Burrage. 2021. Start a conversation with your personal productivity assistant in Outlook with Cortana.

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.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Matt Gardner, William Merrill, Jesse Dodge, Matthew E Peters, Alexis Ross, Sameer Singh, and Noah Smith. 2021. Competency problems: On finding and removing artifacts in language data. *ArXiv preprint*, abs/2104.08646.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12–22, Berlin, Germany. Association for Computational Linguistics.

Michael J Kearns, Umesh Virkumar Vazirani, and Umesh Vazirani. 1994. *An introduction to computational learning theory*. MIT press.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Junyi Jessy Li and Ani Nenkova. 2014. Addressing class imbalance for improved recognition of implicit discourse relations. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 142–150, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Zhuang Li, Lizhen Qu, Shuo Huang, and Gholamreza Haffari. 2021a. Few-shot semantic parsing for new predicates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1281–1291, Online. Association for Computational Linguistics.

Zhuowan Li, Elias Stengel-Eskin, Yixiao Zhang, Cihang Xie, Quan Hung Tran, Benjamin Van Durme, and Alan Yuille. 2021b. Calibrating concepts and operations: Towards symbolic reasoning on real images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14910–14919.

Evan Zheran Liu, Behzad Haghgoo, Annie S. Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021. Just train twice: Improving group robustness without training group information. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 6781–6792. PMLR.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019. Benchmarking natural language understanding services for building conversational agents. In *Proceedings of the Tenth International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, pages xxx–xxx, Ortigia, Siracusa (SR), Italy. Springer.

Petr Lorenc, Petr Marek, Jan Pichl, Jakub Konrád, and Jan Šedivỳ. 2021. Benchmark of public intent recognition services. *Language Resources and Evaluation*, pages 1–19.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Tom M. Mitchell, William W. Cohen, Estevam R. Hruschka Jr., Partha Pratim Talukdar, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, Thahir Mohamed, Ndapandula Nakashole, Emmanouil A. Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard C. Wang, Derry Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, Malcolm Greaves, and Joel Welling. 2015. Never-ending learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2302–2310. AAAI Press.

Toan Q Nguyen and Julian Salazar. 2019. Transformers without tears: Improving the normalization of self-attention. *ArXiv preprint*, abs/1910.05895.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Zdeňka Urešová. 2016. Towards comparability of linguistic graph Banks for semantic parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3991–3995, Portorož, Slovenia. European Language Resources Association (ELRA).

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014a. SemEval 2014 task 8: Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72, Dublin, Ireland. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014b. SemEval 2014 task 8: Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72, Dublin, Ireland. Association for Computational Linguistics.

Yonatan Oren, Shiori Sagawa, Tatsunori B. Hashimoto, and Percy Liang. 2019. Distributionally robust language modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4227–4237, Hong Kong, China. Association for Computational Linguistics.

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.

Emmanouil Antonios Platanios, Adam Pauls, Subhro Roy, Yuchen Zhang, Alexander Kyte, Alan Guo, Sam Thomson, Jayant Krishnamurthy, Jason Wolfe, Jacob Andreas, and Dan Klein. 2021. Value-agnostic conversational semantic parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3666–3681, Online. Association for Computational Linguistics.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *ArXiv preprint*, abs/1911.08731.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. 2020. An investigation of why overparameterization exacerbates spurious correlations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 8346–8356. PMLR.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Semantic Machines, Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, Hao Fang, Alan Guo, David Hall, Kristin Hayes, Kellie Hill, Diana Ho, Wendy Iwaszuk, Smriti Jha, Dan Klein, Jayant Krishnamurthy, Theo Lanman, Percy Liang, Christopher H. Lin, Ilya Lintsbakh, Andy McGovern, Aleksandr Nisnevich, Adam Pauls, Dmitrij Petters, Brent Read, Dan Roth, Subhro Roy, Jesse Rusak, Beth Short, Div Slomin, Ben Snyder, Stephon Striplin, Yu Su, Zachary Tellman, Sam Thomson, Andrei Vorobev, Izabela Witoszko, Jason Wolfe, Abby Wray, Yuchen Zhang, and Alexander Zotov. 2020. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

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

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

Lionel Sujay Vailshery. 2021. Total number of Amazon Alexa skills in selected countries as of January 2021.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey E. Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2773–2781.

Aaron Steven White, Elias Stengel-Eskin, Siddharth Vashishtha, Venkata Subrahmanyan Govindarajan, Dee Ann Reisinger, Tim Vieira, Keisuke Sakaguchi, Sheng Zhang, Francis Ferraro, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2020. The universal decompositional semantics dataset and decomp toolkit. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5698–5707, Marseille, France. European Language Resources Association.

Han-Jia Ye, De-Chuan Zhan, and Wei-Lun Chao. 2021. Procrustean training for imbalanced deep learning. *ArXiv preprint*, abs/2104.01769.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019a. AMR parsing as sequence-to-graph transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 80–94, Florence, Italy. Association for Computational Linguistics.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019b. Broad-coverage semantic parsing as transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3786–3798, Hong Kong, China. Association for Computational Linguistics.

Chunting Zhou, Xuezhe Ma, Paul Michel, and Graham Neubig. 2021. Examining and combating spurious features under distribution shift. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12857–12867. PMLR.

A Data

A.1 Intent recognition

Table 2 contains example utterances for each intent.

A.2 Semantic parsing

The SMCalFlow data consists of user-agent dialogues, where the agent produces executable Lisp programs based on user commands. Variable binding can be performed in Lisp to refer to a value multiple times in a program in a parsimonious way. Underlyingly, the variable binding procedure corresponds to re-entrancy in the DAG encoding the program graph. Thus, the SMCalFlow parsing task can be tackled either at the level of the Lisp string (sequence-to-sequence) or at the level of the DAG (sequence-to-graph), with the latter approach demanding a method for handling re-entrant nodes in a graph.

A.3 Trigger Tokens

Table 3 has the trigger tokens per symbol. These were determined manually by examining tokens which yielded high $\hat{P}(\text{symbol}|t \in \text{input})$ at the lowest data setting.

B Models

B.1 LSTM

The LSTM model takes as input the previous user utterance, the produced agent utterance (if these are available) and the current user utterance, all separated by special tokens. These are tokenized and embedded using an embedding layer initialized with 300-dimensional GloVe embeddings (Pennington et al., 2014). Note that there is no contextualized encoder used here. The encoder is a 2 layer stacked BiLSTM, with a hidden size of 192 and dropout of $p = 0.5$ between cells. The decoder embeddings are initialized randomly and are also 300-dimensional. The decoder also has 2 layers with a hidden size of 384, and recurrent dropout of $p = 0.5$. The source attention is implemented as an MLP with hidden size 64. Batches are bucketed by length during training, and a patience threshold of 20 epochs without improvement is set. The LSTM models are trained with ADAM using a learn rate of $1e-3$ and weight decay of $3e-9$. Note that for SMCalFlow this paradigm is fairly weak due to its tendency to produce malformed Lisp expressions at lower data regimes and the handling of variable binding through `let` expressions.

B.2 Transformer

For the transductive model, the DAG for a program (cf. Fig. 3) is first transformed into a tree by copying and co-indexing re-entrant nodes. The tree is then linearized into a sequence of nodes, edge heads, edge types, and node indices. At test time, the model produces these sequences, which can be deterministically reconstructed into a DAG by merging co-indexed nodes. The generation component of the model maintains a dynamic output vocabulary over three operations: generation from a fixed vocabulary, source copying from the input, and target copying from previously generated tokens. The target copy operation allows the model to handle re-entrant nodes, which appear more than once in the linearized tree. This operation allows us to later recover node indices and thus re-build a DAG by merging copied nodes. The edge heads and labels are parsed by a biaffine parser (Dozat and Manning, 2017). This allows the model to handle functions, arguments, and types separately via typed edges. Each operation type (`ValueOp`, `BuildStructOp`, `CallLikeOp`) corresponds to a different edge type; the edge types for arguments are also indexed to allow for explicit argument ordering (e.g. `arg0`, `arg1`, etc.).

The input to the model is the same as for the LSTM: the concatenation of the previous two dialogue turns, followed by the current user utterance. These are tokenized and embedded with 300-D GloVe embeddings as well as 100-D character CNN features. There is embedding dropout with $p = 0.33$ to prevent overfitting. The input text is also passed through `bert-base-cased`, with each subword receiving a 768-D representation. These are max-pooled across subword tokens to align with the token-level embeddings. The encoder hidden size is 512, with a 8 heads and a feed-forward dimension of 2048. The layer-norm and feedforward layers are swapped, and the weight initialization is downscaled by a factor of 512, following Nguyen and Salazar (2019). The encoder has dropout $p = 0.2$.

For the transformer, the decoder embeddings are also initialized with GloVe and character CNN features. The decoder also has 8 layers with the same dimensions and dropout as the encoder. As in the LSTM model, source attention is implemented as an MLP, here with a hidden dimension of 512. The target attention (for target-side copy) is identical. Source attention uses coverage (See et al.,

Intent	Utterance
play_radio	*play radio mirchi for me*
play_radio	*go to channel one hundred and six point nine*
play_radio	*i want to hear morning edition on npr*
play_radio	*are you set radio on my favorite radio station*
email_query	*open email for unread mails*
email_query	*what is the subject of latest email i got and who sent it*
email_query	*has dad sent any emails recently*
email_query	*new email from mom*
email_querycontact	*find all the contacts named john*
email_querycontact	*what is mary s.'s birthday*
email_querycontact	*what information do you have on file in my information about bill*
email_querycontact	*give me charles telephone number*
general_quirky	*nice to talk to you*
general_quirky	*ask me an arithmetic question*
general_quirky	*i would like it to help with coding debugging*
general_quirky	*i like my robot to talk to me like a friend*
transit_traffic	*what is the traffic situation right in broadway street*
transit_traffic	*what is the traffic like today*
transit_traffic	*is there traffic right now in maiden lane*
transit_traffic	*let me know about current traffic in carmen drive*

Table 2: Examples of intent recognition data.

Symbol	Tokens
email_query	emails, inbox
email_querycontact	contact, phone, number
general_quirky	day, today, tell, can
play_radio	channel, radio, fm, point, station, tune
transport_traffic	traffic
FindManager	boss, manager, supervisor
PlaceHasFeature	takeout, casual, waiter
Tomorrow	tomorrow
FenceAttendee	meet, mom
DoNotConfirm	cancel, n't, no

Table 3: Triggers for each symbol.

2017). The biaffine parser projects the transformer representations to 512 and has dropout $p = 0.2$. The transformer models are trained with a patience of 20, using Adam with a linear learnining rate warmup stage, followed by exponential learning rate decay. We set the number of warmup steps to 8000.

Function	**Dialogue Context**	**Current User Utterance**
`FindManager`	N/A	*Make an event with Abby and her boss*
`FindManager`	**User**: *Who are Jake's reports*, **Agent**: *Jake Cobb has no direct reports.*	*Who does he report to?*
`FindManager`	**User**: *Add an event called presentation with Jamal and his supervisor for Friday at 11.* **Agent**: *Is this good?*	*Add Igor and his supervisor to this as well.*
`Tomorrow`	N/A	*Find an event for tomorrow after 4 pm.*
`Tomorrow`	N/A	*Schedule lunch with Nick tomorrow at noon*
`Tomorrow`	**User**: *What time will the sun rise in seattle tomorrow* **Agent**: *Sunrise will be at 12 : 00 AM tomorrow.*	*what time will the sun set in seattle tomorrow*
`DoNotConfirm`	**User**: *Can you change the time to 4 instead?* **Agent**: *How about now?*	*No, I don't like either of those.*
`DoNotConfirm`	**User**: No I need it to be in the afternoon, **Agent** Does one of these work?	*No they don't*
`DoNotConfirm`	**User**: *Schedule a dentist appointment tomorrow afternoon*, **Agent**: *Does one of these work?*	*No*
`FenceAttendee`	N/A	*Create lunch with mom on sunday*
`FenceAttendee`	**User**: what events do I have tomorrow, **Agent**: I found 2 events tomorrow .	*Add my sister , brother , and Daniel*
`FenceAttendee`	N/A	*Can you tell me if I meet with our repair rep this week or next week ?*
`PlaceHasFeautre`	**User**: What cuisine do they serve ?, **Agent**: Sorry , I can't handle that yet .	*Does the Black Bottle restaurant have a full service bar ?*
`PlaceHasFeature`	**User**: Find me Round Table Pizza in Truckee, **Agent**: I found one option .	*Could I bring a party of people there ?*
`PlaceHasFeature`	N/A	*Is Bamonte 's in Brooklyn capable for large parties ?*

Table 4: Example data for SMCalflow Parsing.