Learning Latent Semantic Annotations for Grounding Natural Language to Structured Data

Guanghui Qin[1]* **Jin-Ge Yao**[2] **Xuening Wang**[3] **Jinpeng Wang**[2] **Chin-Yew Lin**[2]
[1]Peking University, [2]Microsoft Research Asia, [3]University of California Los Angeles
ghq@pku.edu.cn, sherry9788@g.ucla.edu
{jinge.yao, jinpwa, cyl}@microsoft.com

Abstract

Previous work on grounded language learning did not fully capture the semantics underlying the correspondences between structured world state representations and texts, especially those between numerical values and lexical terms. In this paper, we attempt at learning explicit latent semantic annotations from paired structured tables and texts, establishing correspondences between various types of values and texts. We model the joint probability of data fields, texts, phrasal spans, and latent annotations with an adapted semi-hidden Markov model, and impose a soft statistical constraint to further improve the performance. As a by-product, we leverage the induced annotations to extract templates for language generation. Experimental results suggest the feasibility of the setting in this study, as well as the effectiveness of our proposed framework. [1]

1 Introduction

The meaning of natural language should always be accompanied by a context. *Grounded language acquisition* aims at learning the meaning of language in the context of an observed world state. A solution framework typically addresses the following subproblems: segmenting the text into meaningful phrasal units, determining which world state information is being referred to, and finding proper alignments from these units to the events of values in the world state.

The task has attracted much attention from the NLP community with a special focus on aligning text descriptions onto processed, structured event records (Snyder and Barzilay, 2007; Liang et al., 2009; Hajishirzi et al., 2011). Various statistical models have been proposed, attempting at

TEAM	WIN	LOSS	PTS	…	PT-QT4
Raptors	33	15	120	…	31
Wizards	31	17	116	…	15

The Toronto Raptors (33 - 15) barely edged out the Washington Wizards (31 - 17) 120 - 116 in overtime Saturday at the Verizon Center. The Raptors took a great lead before the fourth quarter, but the Wizards fought back by outscoring Toronto 31 - 15 ....

Figure 1: An example of the summary and the corresponding table.

characterizing the interaction between text spans and categorical values (e.g., *direction*=*'East'*) or strings (e.g., *person names*). The previously addressed term *semantic correspondences* narrowly describes the process of aligning natural language spans to different data fields.

However, there still exists a gap between alignment results and the underlying semantics. People tend to use various phrases to describe information that are inferred from different amounts of numerical values in a data field, or values derived from additional operations over fields. Consider the example description for a basketball game shown in Figure 1. The phrase *edged out* in the first sentence implies the fact that *the Toronto Raptors* had beaten their opponent by a relatively narrow margin. This could only be derived from an operation of subtraction between two scores that correspond to the field `PTS` for both teams in the event table, which leads to a relatively small difference of only four points. Previous efforts on learning semantic correspondences relying on *categorical* distributions (Liang et al., 2009) or *string pattern features* (Hajishirzi et al., 2012; Koncel-Kedziorski et al., 2014) do not have the capability to accurately capture *numerical information*, especially for the part that *does not appear explicitly* in the table and needs to be inferred.

* Contribution during internship at Microsoft Research Asia.

[1]Our implementation is available at https://github.com/hiaoxui/D2T-Grounding.

Such kind of language grounding is important both for natural language understanding and for natural language generation. For language understanding, establishing explicit connections between symbols and values beyond ungrounded symbolic meaning representations will be useful for acquisition and inference of numerical commonsense (Narisawa et al., 2013). For language generation, properly aligned information is the key to acquiring patterns of various lexical choices under different world states (Roy and Reiter, 2005).

In this work, we make a step towards more explicit semantic correspondences between structured data and texts. Rather than only producing coarse alignments between data fields and text spans, we try to detect the latent semantics underlying these alignments by prompting explicit semantic annotations. We make the first attempt at utilizing publicly available datasets originally prepared for data-to-text language generation to produce such annotations for words and phrases in natural language without additional supervision.

Specifically, we conduct our study on a recently released dataset of descriptions for NBA basketball games with structured tables of game records. Different from a few popular datasets that have been well conjectured to be produced from rules (Reiter, 2017), the summaries are all written by humans. The text contains some numbers and proper nouns, which are easier to establish correspondence with data. However, the majority of texts contain many informative words, some of which need to be inferred indirectly from various types of values in the data cells. We want to establish explicit correspondences for them.

We derive a set of semantic labels from original data fields (Sec 4.1). These labels could be executed to establish direct correspondences to one or more values in the structured table. No annotation on the original dataset means unsupervised learning from weak distant supervision should be conducted. We design a semi-hidden Markov model to address this problem (Sec 4.2), which could align a semantic label to a word span. In the model, we leverage continuous probability distributions to model the correspondences between numerical values and lexical terms, which has not been well captured in previous work. To address the emerged issue of "garbage collection" that commonly appears in statistical alignment models (Sec 4.5), we add a soft statistical constraint via posterior regularization (Ganchev et al., 2010). As a by-product, we also show how the derived semantic annotations could be used to induce descriptive templates for data-to-text generation (Sec 5). Experimental results (Sec 6) suggest the feasibility of the setting in this study, and show the effectiveness of our proposed framework.

2 Related work

Grounded language acquisition has aroused wide interest in various disciplines (Siskind, 1996; Yu and Ballard, 2004; Gorniak and Roy, 2007; Yu and Siskind, 2013; Chrupała et al., 2015). Later work in the community of natural language processing also moved in this direction by relaxing the amount of supervision to enable a model to learn from ambiguous alignments (Kate and Mooney, 2007; Chen and Mooney, 2008). Some research aimed at establishing coarse alignments between simulated robot soccer game records and commentary sentences (Chen and Mooney, 2008; Chen et al., 2010; Bordes et al., 2010; Hajishirzi et al., 2011). For weather forecast domain, Liang et al. (2009) used a hierarchical hidden Markov model in order to map utterances to world states, which coped with segmentation and alignment together. More recently, Koncel-Kedziorski et al. (2014) tried to obtain the correspondences between real commentaries and structured football (soccer) events in multiple resolutions. We are distinct from this line of work in the fact that we aim at producing explicit semantic annotations that could capture information from structured tables. To achieve this goal, we need additional scaling or operations to enable data fields and values to be faithfully mapped onto texts. This will address the issue of the lack of consideration for the relationship between lexical terms and numerical values. Our approach makes a significant difference in that our framework could generalize to numerical values or value combinations that are unseen in training, and will not be simply reciting cooccurrence patterns of exact values in the training data.

Our work relates to learning executable semantic parsers under weak supervision. Early semantic parsing started from fully supervised training with annotated meaning representations available (Zettlemoyer and Collins, 2005; Ge and Mooney, 2006; Snyder and Barzilay, 2007), but more recent work focused on reducing the amount of supervision required (Artzi and Zettlemoyer, 2013). The intuition behind weakly supervised executable semantic parsing is that once the latent semantic representation has been executed, one could test whether the execution results could match the information with available weak supervision signals such as answers to natural language queries (Clarke et al., 2010; Liang et al., 2011), or task completion from instructional navigations (Misra et al., 2017). Such formulations have been adapted for question answering over structured tables (Pasupat and Liang, 2015; Krishnamurthy et al., 2017). However, the current research focus is to convert a natural language question into executable table queries and to directly retrieve results. They do not have the need of inference involving numerical commonsense implied by various lexical patterns. A few unsupervised approaches exist (Poon and Domingos, 2009; Poon, 2013) but only specific to translating language into queries in the highly structured database and cannot be applied to our domain.

Our approach is implemented as assigning tag annotations over text spans, which is conceptually related to fine-grained named entity tagging (Ling and Weld, 2012). Our setting only requires a rather weak and distant form of supervision from paired tables and texts without annotations for fine-grained alignments between phrases and data cells. Similar modeling and learning strategies could potentially be useful for considerably large tag space derived from structured knowledge bases in the future (Choi et al., 2018).

The feasibility of this work is partly due to the availability of data, mostly comes from the field of data-to-text language generation. Related work in data-to-text generation mainly focused on directly generating summary descriptions for structured data (Mei et al., 2016; Kiddon et al., 2016; Murakami et al., 2017; Wiseman et al., 2017), without establishing underlying semantic correspondences. Texts generated thereby can be fluent but not conforming to the input data, unlike template-based approaches where lexical choices could be directly controlled. In our work, we find the derived semantic correspondences between data and texts to be useful for template induction, either with simple heuristics to automatically extract description patterns (*how to say*) and corresponding triggers (*what & when to say*), or with more crafted discriminative learning approaches (cf. Angeli et al. (2010)).

3 Technical overview

Task Let $\mathcal{S}$ be the set of all world states, $\mathcal{W}$ be the set of all texts, $\mathcal{O}$ be the set of all executable operators, and $\mathcal{V}$ be the output space of $\mathcal{O}$. A world state $s \in \mathcal{S}$ is a table storing some information, or more specifically in this work, a tabular recording for a sports game. An operator $o \in \mathcal{O}$ can be executed on a world state to retrieve values, i.e., each o could be treated as a mapping of $\mathcal{S} \rightarrow \mathcal{V}$. The result of an operation can be a string, a continuous values or a discrete value. Meanwhile, each world state s is accompanied with a piece of description $\mathbf{w} \in \mathcal{W}$. Here $\mathbf{w}$ consists of a sequence of word tokens $\{w_i \in \mathbf{w}\}$. We further define a segmentation variable π, which could convert $\mathbf{w}$ into a sequence of word spans $\mathbf{c}$, containing each span of tokens $c^t \in \mathbf{c}$ that could be interpreted as a phrase. Note that we use superscript t to denote indices of phrases, and subscript i to denote indices of individual words. We further define $\mathbf{l}$ as a sequence of latent labels, and value of each l^t is an operator $o_i \in \mathcal{O}$. [2] For each world state s and corresponding description $\mathbf{w}$, we want to jointly find a proper segmentation π to obtain $\mathbf{c}$, and assign labels to every word span c^t.

We conduct this study on the RotoWire subset of the openly available dataset released by Wiseman et al. (2017), containing text descriptions for NBA basketball games with structured tables of game statistics. Take Figure 1 as an example. The proper nouns (e.g. *Toronto Raptors*) appeared in the sentence can be assigned with a tag `Team_Name` in our tag set, which could then be aligned to the *Team Name* field in the table. Some numbers appearing in the text, such as *15* in the example, can be assigned with the label `Team_Losses`, with the executable annotation to extract the number of previously lost matches of the mentioned team. What we are more interested in is where the phrase *edge out* comes from. We are aiming at a model which is capable to capture the semantics behind the phrase *edge out*, which is used to describe an event that one team has beaten the other with close scores. In our annotation scheme, this phrase should be assigned with the tag `Team_Points_Delta`, and executing that will return the score difference between two teams.

The task is challenging in that there does not

[2]We will interchangeably use *labels*, *operators*, *tags* to refer to the latent executable semantic annotations in this paper.

Output Type	Example
String	`Team_City`
Categorical	`Player_Start_Position`
Numerical	`Team_Points_Delta`

Table 1: Examples from the derived tag set, where each tag could correspond to one of three types of values.

exist any other additional supervision signal. The learning process will mostly rely on statistical cooccurrences of information between the structured data and its text descriptions. Note that we assume a consistent structure (schema) throughout the whole dataset upon which the learning process will be performed.

Model To jointly learn word segmentations $\mathbf{c}$ and latent semantic annotations $\mathbf{l}$ between world state s and text $\mathbf{w}$ in a unified framework, we propose a generative model to characterize the joint distribution $P_s(\mathbf{l}, \pi, \mathbf{w}; \theta)$, parameterized by θ.

Learning The data contain paired texts and tables only, thus our model must learn segmentations and latent semantic annotations in an unsupervised fashion. The target is to maximize the complete data likelihood

$$\mathcal{L}(\theta) = \prod_{(s,\mathbf{w}) \in \mathcal{D}} \sum_{\mathbf{l},\pi} P_s(\mathbf{l}, \pi, \mathbf{w}; \theta),$$

where $\mathcal{D}$ represents the whole training data. To reduce the search space in inference and to capture some patterns of content planning in the text descriptions, we adopt a Markov assumption over phrase segments, which leads to a hidden semi-Markov model (semi-HMM) (Murphy, 2002; Sarawagi and Cohen, 2005). The key part is to characterize different types of correspondences (Sec 4.2). We derive an expectation-maximization (EM) algorithm to perform maximum likelihood estimation, and introduce a soft statistical regularization to guide the model towards a better solution (Sec 4.5).

Inference Once the model has been trained, we use a Viterbi-like dynamic programming process to perform MAP inference to segment the texts and to assign the most likely tags for each span.

4 Framework

4.1 The set of annotations

We describe the process of how we derived our set of semantic annotations here. There are two kinds of specific tables for each NBA game in the dataset: *Box-Scores* and *Line-Scores*, respectively showing the performance statistics for individual players and the whole teams. The types of possible values for data fields are strings, categorical values and numerical values. *Box-Score* consists of 24 fields, of which four are string-values, one is categorical, the other 19 being numerical. *Line-Score* consists of 16 fields, containing two string-valued, one categorical, and 13 numerical fields. A semantic tag works on a specified kind of field type, from either team statistics or player statistics. For a tag that could align to one single field in the table, we let it return the exact value in the cell that could maximize the likelihood. For example, the tag `Team_City` is used to extract the name of the team in the table that corresponds to the current word span. For fields taking numerical values, we additionally allow tags to be able to perform mathematical operations, such as subtraction, between two values in the same field. [3] For example, the tag `Team_Points_Delta` can be executed to return the score difference between two teams. We list the different types of tags with examples shown in Table 1 and leave the entire tag set to Appendix A. Along with all these tags derived from the original data fields, we also include a special `NULL` tag which are supposed to be assigned to non-informative words or words containing information not contained in the given table.

Note that we impose little prior knowledge in this step. We simply over-generate all possible labels, and let the model figure out which part of them should be eventually used. Although the only compositional operation we used in this work is numeric subtraction, common operations that could produce string, categorical values or numbers could be easily introduced for other domains.

4.2 Semi-HMMs with continuous values

As previously mentioned, we will be modeling the joint distribution of word segmentation $\mathbf{c}$ and the latent semantic annotations $\mathbf{l}$ between paired world state s and text $\mathbf{w}$, which could be factorized as:

$$P_s(\mathbf{l}, \pi, \mathbf{w}) = P_s(\mathbf{w}, \pi | \mathbf{l}) \cdot P_s(\mathbf{l}),$$

and we write $P_s(\mathbf{w}, \pi|\mathbf{l})$ as $P_s(\mathbf{c}|\mathbf{l})$. In this section, we focus on the probability of the alignments between word spans and labels, namely, $P_s(\mathbf{c}|\mathbf{l})$.

Following Liang et al. (2009), we consider two aspects. One is *salience* that captures the intuition

[3] We limit the number of arguments within two in this work and leave more complex operators for future study.

that some fields should be more frequently mentioned than others (henceforth some latent tags should be more frequently triggered). The other is (local) *coherence*, which refers to the order in which the writer mentioning certain information tends to follow some patterns. To capture these two phenomena, we define a Markov model:

$$P_s(\mathbf{c}, \mathbf{l}) = \prod_t P(l^t | l^{t-1}) \cdot P_s(c^t | l^t),$$

where l^t is the annotated label at time stamp t, and we assume that the transition probabilities are independent of world state s. It resembles a standard form of HMMs, despite the subscript s in $P_s(\mathbf{c}|\mathbf{l})$. For different types of correspondences between l^t and c^t, we define different probability distributions to model $P_s(c^t|l^t)$:

(1) **Numerics-to-numerics**: The numbers in texts could sometimes be inaccurate due to some rounding customs, thus we use a Gaussian model for this type: $\mathit{SoftIndicator}(x, y|\sigma) = \mathcal{N}(x - y|0, \sigma)$, where $\mathcal{N}$ is the Gaussian density. When the output type of tag l^t is numeric and the word span c^t is a number, we set $P_s(c|l) = \mathit{SoftIndicator}(c, v_l|\sigma_l)$, where σ_l is different for different tags, and v_l is the corresponded value in the table for tag l. Note that when $\sigma \to 0$, $SoftIndicator$ reduces to an indicator function that only allows exact matching.

(2) **String-to-string**: Similarly for strings, since simple matching could fail if the text contains *Bob* to refer to *Bob Smith*. We simply use string matching to model the probability: $P_s(c|l) \propto \mathit{Match}(v_l, c)$, where the *Match* function returns the number of shared words between cell value v_l and word c.

(3) **Category-to-string**: For labels correspond to discrete categorical values, such as `Sunday`, `PG` (*point guard*, a basketball position), we adopt the same method used by Liang et al. (2009): using a multinomial distribution over all word spans for each possible category:

$$P_s(c|l) = \nu_{c,v_l}, \quad \sum_c \nu_{c,v} = 1, \qquad (1)$$

where v_l is again the output value of tag l.

(4) **Numerics-to-string**: When the tag corresponds to a numeric value v_l while the word span c is not a number, the problem resembles speech modeling (Huang et al., 1990). Applying the Bayes rule, we get: [4]

$$P_s(c|l) = P(c|l, v_l) \propto P(v_l|c, l) \cdot P(c|l),$$

where we collapse the relevant part from world state s into v_l. The intuition behind $P(v_l|c, l)$ is that when an informative word (e.g. *routed*) appears in the text, the corresponding values should have different chances to happen in the world, e.g. $P(30|\mathit{routed}, \texttt{Points\_Delta}) > P(3|\mathit{routed}, \texttt{Points\_Delta})$.

Due to the lack of prior knowledge on the distribution, we also model this term as Gaussian. [5] The result resembles a Gaussian mixture:

$$P_s(c|l) \propto \mathcal{N}(v_l; \mu_{c,l}, \sigma_{c,l}) \cdot \eta_{c,l},$$
$$\sum_c \eta_{c,l} = 1,$$

where $P(c|l) = \eta_{c,l}$ is also multinomial.

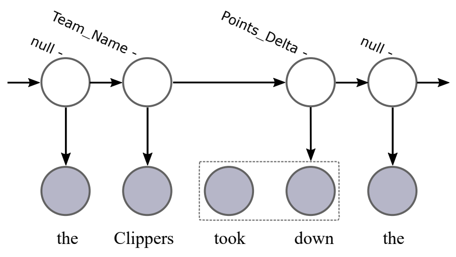


Figure 2: A Semi-HMM that can yield an entire phrase from one tag.

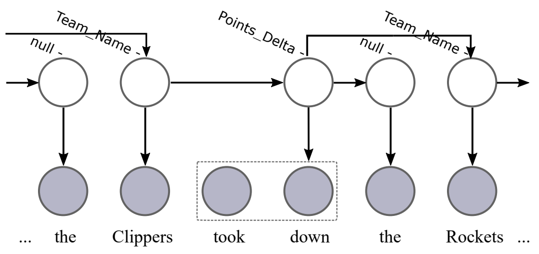


Figure 3: A slightly different Semi-HMM whose transition score is calculated by skipping `NULL` fields.

4.3 Modeling phrasal spans

Our model can enable phrase segmentation. Previously, Liang et al. (2009) treated the words inside a phrase individually and independently. This

[4]We noticed that in parallel with our work, another study (Zhang et al., 2018) on verb selection for data-to-text generation also use the same strategy of Bayes rule to estimate parameters, and provide a noisy-channel interpretation.

[5]The double tails in Gaussian pdf have different interpretations: unlikely to occur, or unlikely to occur conditionally.

could be problematic in our scenario. For example, *take down* is used to describe a team defeating another, while separately both *take* and *down* are frequent words in general, making them difficult to be jointly assigned with the correct label as a whole. Instead, in our model we treat the phrasal word span as a unit. The probability is assigned to the whole span of phrase instead of individual words, which will break the token-wise Markov property (Fig. 9, henceforth Semi-HMM). For efficient parameter estimation, We use a variant of the standard forward-backward algorithm by adding a parameter k, which is the maximum length of word spans, onto the Markov chain. We leave detailed descriptions to Appendix B.

4.4 Skipping null labels

Preliminary experiments suggest that the initial model have too many words assigned to the `NULL` tag. Informative alignments may not be adjacent, which breaks the simplest Markov assumptions. In our model, the transition score of two non-`NULL` labels can be calculated by skipping all the `NULL`s in between, as shown in Figure 3. This is implemented without breaking the overall Markov property with the following trick used in earlier work on statistical alignments (Brown et al., 1993): Suppose we have m labels (i.e., m latent states), we can design m different `NULL` labels that share the same emission score, while preserving their original outward-transition probabilities. The types of `NULL` labels are inherited from the previous label. This might seem to be wasteful at first sight as we use two-fold latent states, but the Markov property is successfully preserved, therefore simplifying our implementation.

4.5 Posterior regularization

The structured tables and text descriptions of the dataset were originally crawled from different sources. As a consequence, a non-negligible proportion of texts is in fact describing information outside the given table, such as historical records (e.g., *"win streak"*). Ideally, words in these parts of the text should remain unaligned, or in our setting, be annotated with the `NULL` tag. However, due to the notorious effect of *garbage collection* from statistical alignment (Brown et al., 1993), these words tend to be aligned to some irrelevant fields in the table which are rarely mentioned.

We address this issue by adding a soft statistical constraint in the form of posterior regularization (Ganchev et al., 2010; Graça et al., 2010). With posterior regularization, we could add certain types of statistical constraints to the E-step in the EM procedure, while keeping the inference tractable. The constraints should be in the form of:

$$\mathbb{E}[\mathbf{f}(\mathbf{w},\mathbf{l})] = \sum_i \mathbb{E}[\mathbf{f}(\mathbf{w},l_i)] \le \mathbf{b}_{\mathbf{w}}, \quad (2)$$

where the features $\mathbf{f}$ should be defined on local cliques for tractable inference.

We use projected gradient descent to solve the E-step sub-problem in this work. The statistical constraint we add to the posterior is rather simple: For each sentence, we "encourage" at least a proportion of words to be aligned to `NULL` labels:

$$\mathbb{E}[-f(\mathbf{w},\mathbf{l})] \le -r_0 \cdot n, \quad (3)$$

$$f(\mathbf{w},\mathbf{l}) = \sum_{i=1}^{n} \mathbb{1}(l_i = \texttt{NULL}), \quad (4)$$

where r_0 is a adjustable ratio, n is the length of $\mathbf{w}$. We also tried other constraints but found this simple soft regularization performing well.

5 By-product: template induction

Intuitively, the assigned semantic correspondences could be useful to derive templates and trigger rules for language generation. In this work, we use the most straightforward heuristics to perform template induction, utilizing the established correspondences and inferred parameters. Specifically, we first blank out the correspondences of numerics-to-numerics and string-to-string to be empty slots and replace with the tag names. In the example of Figure 1, we could replace *Raptors* with `Team_Name`, and 120 with `Team_Points`, if they have been correctly aligned.

We also need to know when to use each template. We define a *template trigger* to be a quadruple $(c, l, \mu_{c,l}, \sigma_{c,l})$, where c is a phrase, l is a tag, $\mu_{c,l}$ and $\sigma_{c,l}$ are estimated Gaussian parameters. [6] We assign each template with a score to be the minimum probability for all triggers inside:

$$\text{score}(s, \mathbf{t}) = \min_i \mathcal{N}(t_i.l(s); t_i.\mu, t_i.\sigma), \quad (5)$$

where $\mathbf{t} = \{t_i\}$ denotes all possible triggers in the template, and the tag l can be executed over the world state s to retrieve a value $l(s)$. We only

[6]To use a unified notation, for categorical-value triggers we set $\mu_{c,l} = \arg\max_{v_l} \nu_{c,v_l}$ (defined in (1)) and $\sigma_{c,l} = \epsilon$.

consider sentences satisfying both of the following conditions in order to extract templates with high quality: (1) sentences aligned to $\leq$ two teams or one player, and (2) sentences with triggers derived from continuous distributions.

Now that the templates and triggers are ready for use, we will experiment with the following straightforward rules to perform data-to-text generation: For every game, we first generate a sentence describing the scoreline result, followed by three sentences describing other information about team performance. While keeping that no template is repeatedly used, we will then choose the template with the highest score for top ten players sorted by their game points.

6 Experiments

6.1 Experimental setup

We conducted experiments on the ROTOWIRE subset of the Wiseman et al. (2017) dataset. In our experiment, we restricted the maximum length of word span to two as a trade-off of speed and performance. Empirically, most of the phrases in the dataset are at a length of at most two. We empirically set the expected `NULL` ratio to be $r_0 = 0.5$.

We did the following pre-processing steps for all systems in comparison: we lemmatized all tokens in the sentences, and filtered out sentences containing less than five words since they are meaningless short sentences. To utilize the game dates, we converted them from calendar date to the day of week, e.g. 11/28/2016 is converted to *Monday* as a categorical value. Due to the huge noise in the ROTOWIRE dataset, containing many sentences irrelevant with their corresponding tables, we filtered out the sentences that contain no team or player names, or those that mention more than 2 players, as most of them are irrelevant texts.

Following Liang et al. (2009), we also used the parameters of a simpler model without Markov dependency (which was uniformly initialized) to initialize our complete model with obtained parameters, and then trained it until convergence. We adapted Liang et al. (2009)'s framework to the table schema in the ROTOWIRE dataset, and ran experiments accordingly as our baseline model.

6.2 Evaluation

6.2.1 Intrinsic evaluation

It is difficult to evaluate the accuracy of tag assignments for the entire dataset, since the tags are not annotated in the original data. We recruited three human annotators with familiarity in the domain of basketball games to label 300 sentences (around 8,000 tokens in total, and 30% of them are annotated with tags) from the test set. There exists a fraction (18%) where agreements were not made, we included all the proposed tags from the annotators to be correct. Also, because of the ambiguity of annotations, we use the base names of derived tags (e.g. `Rebounds_Delta`) for numerics-to-string relationship evaluation. (e.g. `Rebounds_Delta` is reduced to `Rebounds`) Finally, we calculated the precision and recall for non-`NULL` tag assignments at word-level.

	Precision	Recall	F1
Liang09	0.319	0.643	0.426
Liang09+PR	0.397	0.640	0.490
Semi-HMMs	0.254	0.765	0.381
Semi-HMMs+PR	**0.504**	**0.786**	**0.614**

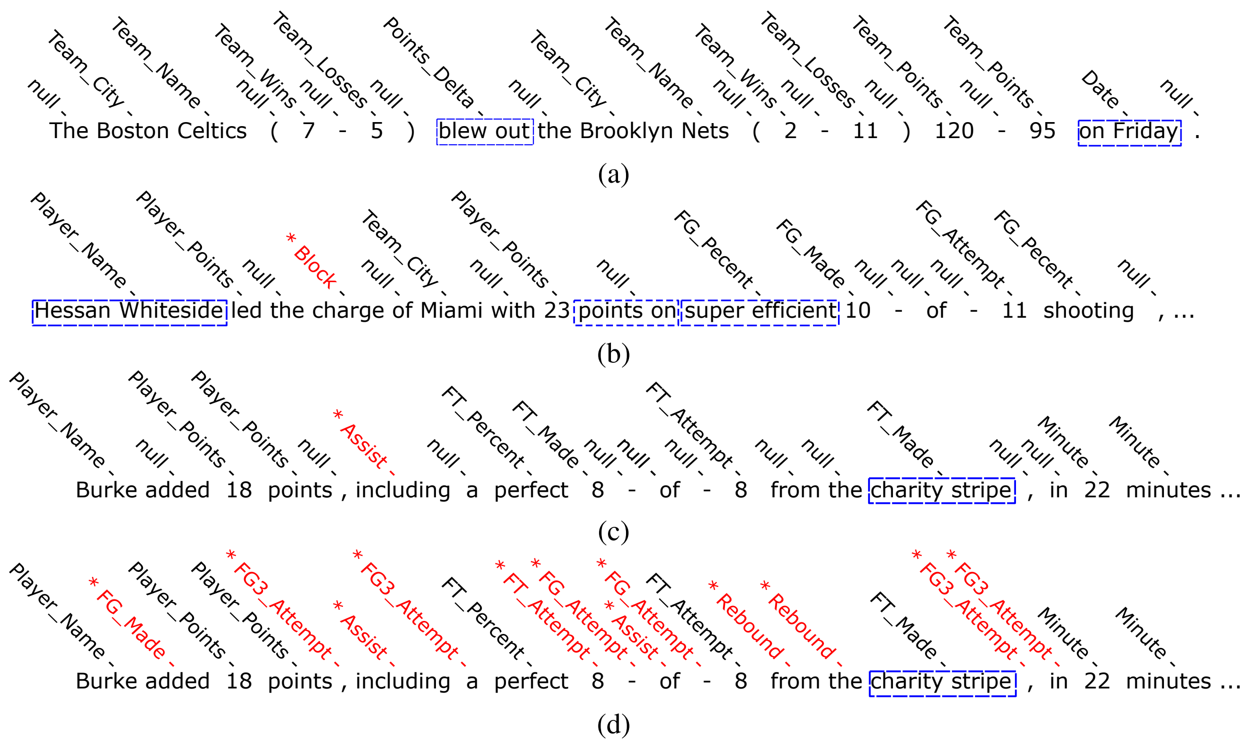
Table 2: Word-level tag assignment results.

Correspondence type	Proportion	Accuracy
numerics-to-numerics	0.369	0.950
numerics-to-string	0.283	0.402
string-to-string	0.252	0.892
category-to-string	0.095	0.936

Table 3: The performance of Semi-HMMs+PR for different types of correspondences.

The results are shown in Table 2. The Liang et al. (2009) framework could still achieve around 65% recall, because there exist a large proportion of correspondences that could be easily captured by exact matches and simple categorical distributions. Our model without PR achieves lower precision than the baseline, because the baseline did not model numerics-to-string relationships and encountered less severe issues of garbage collection. We can observe that our initial model indeed outperforms the baseline system in recall, while PR helps a lot to avoid distraction from irrelevant information that should be tagged as `NULL`.

We also include more fine-grained results for different types of correspondences, shown in Table 3. As expected, numerics-to-string correspondences are the most difficult part in this study. Another notable thing is that although we found that around 40% of numerics-to-numerics correspondences were ambiguous due to the appear-



The Boston Celtics (7 - 5) blew out the Brooklyn Nets (2 - 11) 120 - 95 on Friday .
null Team_City Team_Name null Team_Wins null Team_Losses null Points_Delta null Team_City Team_Name null Team_Wins null Team_Losses null Team_Points null Team_Points Date null
(a)
Hessan Whiteside led the charge of Miami with 23 points on super efficient 10 - of - 11 shooting , ...
Player_Name Player_Points null * Block null Team_City null Player_Points null FG_Pecent FG_Made null null null FG_Attempt FG_Pecent null
(b)
Burke added 18 points , including a perfect 8 - of - 8 from the charity stripe , in 22 minutes ...
Player_Name null Player_Points Player_Points null * Assist null FT_Percent FT_Made null null FT_Attempt null null null FT_Made null null Minute Minute
(c)
Burke added 18 points , including a perfect 8 - of - 8 from the charity stripe , in 22 minutes ...
Player_Name * FG_Made Player_Points Player_Points * FG3_Attempt * Assist * FG3_Attempt FT_Percent * FT_Attempt * FG_Attempt * Assist * FG_Attempt FT_Attempt * Rebound * Rebound FT_Made * FG3_Attempt * FG3_Attempt Minute Minute
(d)

Figure 4: Example of latent tag assignment. Words in blue dashed boxes are phrases recognized by our model. The labels red with asterisks (*) are false assignments. (a), (b), (c) are example of Semi-HMMs-PR, while (d) is produced from the initial Semi-HMMs.

ance of identical values from different table cells, our model could still achieve a high accuracy of 95.0%.

6.2.2 Extrinsic evaluation

We also tested how the derived templates could perform in language generation, when compared with the baseline using the same heuristics described in Sec 5. We report automatic metrics including BLEU scores and those based on relation extraction as proposed by Wiseman et al. (2017): precision & number of unique relations in generation (RG), precision & recall for content selection (CS), and content ordering (CO) score. These automatic metrics were designed for various aspects in NLG and may not all suit our main focus well, so we also conducted human evaluation on information correctness (1-5 scale ratings, the higher the better). We asked four human raters who are fluent in English and with familiarity in basketball terms to rate over outputs for 30 random games. Results are shown in Table 4. We can observe that templates derived from our model indeed outperform those from the baseline.

We put some inducted templates and generated text examples in the Appendix.

6.3 Analysis

Figure 4 shows some examples produced from our methods. Some of the alignments are meaningful, for example, the model assigned the word *perfect* with the annotation `FT_Percent`, which represents the percentage of free throws. Without PR, our model performed poorly by aligning many common words to those rarely mentioned cells. In this example, the `FT_Made` and `FT_Attempt` fields in the input table both have the same value 8, making it difficult for a model without proper local coherence modeling to distinguish between them. Because our initial model without PR cannot annotate `NULL` correctly, the Markov transition between these two numbers was intercepted by three meaningless tokens. However, after injecting the PR constraint, most of the unmentioned words were successfully identified. The model captured the pattern that `FT_Attempt` almost always follows `FT_Made`, making it correctly assigned these two labels.

We conducted ablation experiments for some of the components (Table 5). When setting the maximum phrase length to be $k = 1$, the model degenerates to a normal HMM. The performance measured by F1-score drops for only a little. One

Model	RG(P%)	RG(#)	CS(P%)	CS(R%)	CO	BLEU	Correctness
Liang09+PR	85.83	33.29	14.33	31.09	6.25	8.34	2.60
Semi-HMMs+PR	90.47	41.79	21.63	50.17	9.63	9.45	3.58
Gold-standard	91.77	12.84	100	100	100	100	4.88

Table 4: Results for data-to-text generation (Kendall's W=0.83 from correctness raters)

possible reason is that Semi-HMMs tend to output some meaningless combinations of words as phrases, such as the phrase *points on* in Figure 4 (b), which could lead to many redundant annotations that hampers precision albeit its help to recall. We also tried to disable the transition probabilities during both training and inference, which led to lower precision and lower recall naturally as there was no modeling for local coherence. Finally, by canceling the NULL-skipping mechanism, we found that the numerics-to-numerics annotation accuracy dropped from 95.0% to 88.8%. Many of the spurious numerics-to-numerics annotations, such as the *8* - *of* - *8* in Figure 4 (d), could be corrected using transition probabilities under the skipping-NULL mechanism (Figure 4 (c)).

	Precision	Recall	F1
Semi-HMMs+PR	0.504	0.786	0.614
HMM+PR	0.545	0.694	0.610
No transition	0.468	0.633	0.538
No skip	0.454	0.737	0.562

Table 5: Ablation results.

One additional advantage of our model is that we can easily verify what the model has captured. For the latent annotation `Team_Points_Delta`, we sort its corresponding phrases by weights $\frac{P(c|l)}{P(c)}$ and we list the top 12 weighted words in Figure 5. We can observe that most of the displayed phrases have strong semantic relationship with score differences. More interestingly, we found the mean and variance values estimated by the Gaussian distributions rather informative. When l = `Teams_Points_Delta`, we observed that $\mu_{l,\text{narrowly}} \approx 2$ while $\mu_{l,\text{blow out}} \approx 26$. We could infer the conditions under which some phrases should be used, providing useful insights for lexical choices in language generation.

7 Conclusion

In this paper, we attempt to learn executable latent semantic annotations from paired structured tables and texts. We model the joint probability of

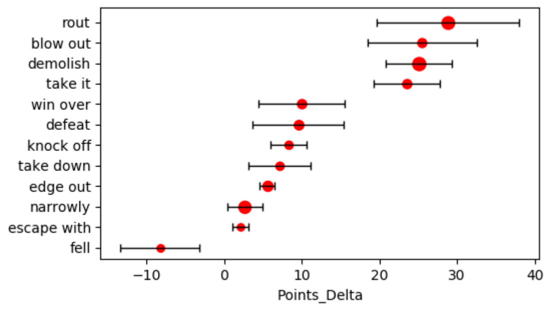


Figure 5: Mean ± std for the top 12 weighted phrases assigned with `Team_Points_Delta`. (circle sizes are proportional to weights)

data fields, texts, phrasal spans, and latent annotations with an adapted semi-hidden Markov model and impose a soft statistical constraint via posterior regularization. Experimental results suggest the feasibility of the setting in study and the effectiveness of our framework.

This is a preliminary study for using weak supervision from structured data and texts to address the challenging problem of language grounding. For future study, one could collect large-scale data and texts in other domains where more complex grounding on phrases such as "increasing trends" should be done. To enhance modeling power, unsupervised discriminative models that utilize rich features (Berg-kirkpatrick et al., 2010) could also be explored. We are also interested in collecting more high-quality parallel data to induce grounded compositional logic representations.

Acknowledgements

We would like to thank Hongyuan Mei and all the anonymous reviewers for giving helpful comments on an earlier draft of this paper.

References

G. Angeli, P. Liang, and D. Klein. 2010. A Simple Domain-Independent Probabilistic Approach to Generation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Y. Artzi and L. S. Zettlemoyer. 2013. Weakly Supervised Learning of Semantic Parsers for Mapping Instructions to Actions. *Transactions of the Association for Computational Linguistics (TACL)*.

T. Berg-kirkpatrick, A. Bouchard-Côté, J. DeNero, and D. Klein. 2010. Painless unsupervised learning with features. In *North American Association for Computational Linguistics (NAACL)*.

A. Bordes, N. Usunier, and J. Weston. 2010. Label Ranking under Ambiguous Supervision for Learning Semantic Correspondences. In *International Conference on Machine Learning (ICML)*.

P. F. Brown, S. A. D. Pietra, V. J. D. Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*.

D. L. Chen, J. Kim, and R. J. Mooney. 2010. Training a multilingual sportscaster: Using perceptual context to learn language. *Journal of Artificial Intelligence Research (JAIR)*.

D. L. Chen and R. J. Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *International Conference on Machine Learning (ICML)*.

E. Choi, O. Levy, Y. Choi, and L. Zettlemoyer. 2018. Ultra-fine entity typing. In *Association for Computational Linguistics (ACL)*.

G. Chrupała, Á. Kádár, and A. Alishahi. 2015. Learning language through pictures. In *Association for Computational Linguistics (ACL)*.

J. Clarke, D. Goldwasser, M. Chang, and D. Roth. 2010. Driving Semantic Parsing from the World's Response. In *Computational Natural Language Learning (CoNLL)*.

Michael Collins. 2013. The Forward-Backward Algorithm (lecture notes).

K. Ganchev, J. V. Graça, J. Gillenwater, and B. Taskar. 2010. Posterior Regularization for Structured Latent Variable Models. *Journal of Machine Learning Research (JMLR)*.

R. Ge and R. J. Mooney. 2006. Discriminative reranking for semantic parsing. In *Association for Computational Linguistics (ACL)*.

P. Gorniak and D. Roy. 2007. Situated language understanding as filtering perceived affordances. *Cognitive Science*.

J. V. Graça, K. Ganchev, and B. Taskar. 2010. Learning Tractable Word Alignment Models with Complex Constraints. *Computational Linguistics*.

H. Hajishirzi, J. Hockenmaier, Erik T. Mueller, and E. Amir. 2011. Reasoning about RoboCup Soccer Narratives. In *Uncertainty in Artificial Intelligence (UAI)*.

H. Hajishirzi, M. Rastegari, A. Farhadi, and J. Hodgins. 2012. Semantic Understanding of Proefessional Soccer Commentaries. In *Uncertainty in Artificial Intelligence (UAI)*.

X. D. Huang, Y. Ariki, and M. A. Jack. 1990. *Hidden Markov Models for Speech Recognition*. Edinburgh University Press.

R. J. Kate and R. J. Mooney. 2007. Learning language semantics from ambiguous supervision. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

C. Kiddon, L. Zettlemoyer, and Y. Choi. 2016. Globally Coherent Text Generation with Neural Checklist Models. In *Empirical Methods in Natural Language Processing (EMNLP)*.

R. Koncel-Kedziorski, H. Hajishirzi, and A. Farhadi. 2014. Multi-Resolution Language Grounding with Weak Supervision. In *Empirical Methods in Natural Language Processing (EMNLP)*.

J. Krishnamurthy, P. Dasigi, and M. Gardner. 2017. Neural Semantic Parsing with Type Constraints for Semi-Structured Tables. In *Empirical Methods in Natural Language Processing (EMNLP)*.

P. Liang, M. I. Jordan, and D. Klein. 2009. Learning Semantic Correspondences with Less Supervision. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

P. Liang, M. I. Jordan, and D. Klein. 2011. Learning Dependency-Based Compositional Semantics. In *Association for Computational Linguistics (ACL)*.

X. Ling and D. S. Weld. 2012. Fine-Grained Entity Recognition. In *AAAI Conference on Artificial Intelligence (AAAI)*.

H. Mei, M. Bansal, and M. R. Walter. 2016. What to talk about and how? Selective Generation using LSTMs with Coarse-to-Fine Alignment. In *North American Association for Computational Linguistics and Human Language Technology (NAACL-HLT)*.

D. Misra, J. Langford, and Y. Artzi. 2017. Mapping Instructions and Visual Observations to Actions with Reinforcement Learning. In *Empirical Methods in Natural Language Processing (EMNLP)*.

S. Murakami, A. Watanabe, A. Miyazawa, K. Goshima, T. Yanase, H. Takamura, and Y. Miyao. 2017. Learning to Generate Market Comments from Stock Prices. In *Association for Computational Linguistics (ACL).*

K. P. Murphy. 2002. Hidden Semi-Markov Models (HSMMs).

K. Narisawa, Y. Watanabe, J. Mizuno, N. Okazaki, and K. Inui. 2013. Is a 204 cm Man Tall or Small? Acquisition of Numerical Common Sense from the Web. In *Association for Computational Linguistics (ACL).*

P. Pasupat and P. Liang. 2015. Compositional Semantic Parsing on Semi-Structured Tables. In *Association for Computational Linguistics (ACL).*

H. Poon. 2013. Grounded Unsupervised Semantic Parsing. In *Association for Computational Linguistics (ACL).*

H. Poon and P. Domingos. 2009. Unsupervised semantic parsing. In *Empirical Methods in Natural Language Processing (EMNLP).*

E. Reiter. 2017. You Need to Understand your Corpora! The Weathergov Example.

D. Roy and E. Reiter. 2005. Connecting language to the world. *Artificial Intelligence.*

S. Sarawagi and W. W. Cohen. 2005. Semi-Markov Conditional Random Fields for Information Extraction. In *Advances in Neural Information Processing Systems (NIPS).*

J. M. Siskind. 1996. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition.*

B. Snyder and R. Barzilay. 2007. Database-text alignment via structured multilabel classification. In *International Joint Conference on Artificial Intelligence (IJCAI).*

S. Wiseman, S. M. Shieber, and A. M. Rush. 2017. Challenges in Data-to-Document Generation. In *Empirical Methods in Natural Language Processing (EMNLP).*

C. Yu and D. H. Ballard. 2004. On the Integration of Grounding Language and Learning Objects. In *Association for the Advancement of Artificial Intelligence (AAAI).*

H. Yu and J. M. Siskind. 2013. Grounded Language Learning from Video Described with Sentences. In *Association for Computational Linguistics (ACL).*

L. S. Zettlemoyer and M. Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *Uncertainty in Artificial Intelligence (UAI).*

D. Zhang, J. Yuan, X. Wang, and A. Foster. 2018. Probabilistic Verb Selection for Data-to-Text Generation. *Transactions of the Association for Computational Linguistics (TACL).*

Appendices

A Tag Set

Here we list all the tags we used in our model, which were derived from the table schema in the Wiseman et al. (2017) ROTOWIRE dataset.

- Date
- Team_City
- Team_Name
- Team_PTS (PTS = Points)
- Team_Wins (historical data)
- Team_Losses (historical data)
- Team_QT1_PTS (QT = Quarter)
- Team_QT2_PTS
- Team_QT3_PTS
- Team_QT4_PTS
- Team_FT_Percent (FT = Free Throw)
- Team_FG_Percent (FG = Field Goal)
- Team_FG3_Percent (FG3 = 3-Points Goal)
- Team_AST (AST = Assists)
- Team_REB (REB = Rebounds)
- Team_TOV (TOV = Turnover)
- Team_Wins_Delta
- Team_Losses_Delta
- Team_QT1_PTS_Delta
- Team_QT2_PTS_Delta
- Team_QT3_PTS_Delta
- Team_QT4_PTS_Delta
- Team_FT_Percent_Delta
- Team_FG_Percent_Delta
- Team_FG3_Percent_Delta
- Team_AST_Delta
- Team_REB_Delta
- Team_TOV_Delta
- Player_Name
- Player_City
- Player_Start_Position
- Player_Minute
- Player_PTS
- Player_TOV
- Player_REB
- Player_AST
- Player_DREB (D = Defensive)
- Player_OREB (O = Offensive)
- Player_PF (PF = Personal Fouls)
- Player_FGM (M = Made)
- Player_FGA (A = Attempt)
- Player_FG_Percent
- Player_FG3M
- Player_FG3A
- Player_FG _Percent
- Player_FTM
- Player_FTA
- Player_FT_Percent

B Adapted Forward-Backward algorithm for Semi-HMMs

In this section, we show that in the settings where a latent state could emit a word span longer than one token, we could adapt the forward-backward algorithm to calculate the expectations, with a limited maximum length similar to what is used by Sarawagi and Cohen (2005).

We adopt the notation similar to Collins (2013). We use superscript t as the index of word spans, and subscript i as the index of individual words. We use w_i to represent i-th individual word, and c^t as t-th word span. For example, given the following sentence

$$\underline{\text{The}}\ \underline{\text{quick brown fox}}\ \underline{\text{jumps}}\ \underline{\text{over}}\ \underline{\text{a}}\ \underline{\text{lazy dog}}$$

, where underlines separate different word spans. In this example, there are 9 individual words and 6 word spans, e.g. w_3 = *brown* and c^2 = *quick brown fox*.

We further define $i(t)$ as the index of the last word of word span c^t, $t(i)$ as the index of word span ended with word w_i, $c(i,k)$ as the word span which ends with word w_i with length k, and k^t as the length of c^t. E.g., in the above sentence, we have $i(t=2)=4$, $t(i=4)=2$, $c(i=4,k=3)$ = *quick brown fox* and $k^2=3$.

Note that the tags are annotated on word spans instead of individual words, so we use l^t to represent the latent state of word span c^t.

Let $t(l^t|l^{t-1})$ be the transition probability from latent state l^{t-1} at time $(t-1)$ to l^t at time t, and $e(c|l)$ be the emission probability from latent state l to word span c. To simplify our notation, we define a potential function as the following:

$$\psi(i,k,l',l) = t(l'|l) \cdot e(c(i,k)|l').$$

Figure 6 shows an example. With the potential function defined above, the joint probability of the whole sentence can be written as

$$\psi(\mathbf{c},\mathbf{l}) = \prod_t \psi(i(t),k^t,l^t,l^{t-1}),$$
$$c^{t(i)} = [w_{i-k+1},\ldots,w_i]$$

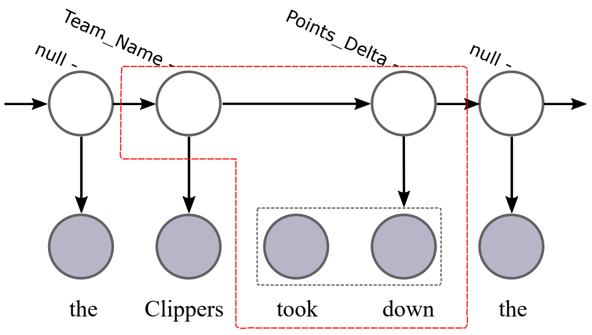


Figure 6: Example of a local potential function $\psi(i,k,l',l) = t(l'|l) \cdot e(c(i,k)|l')$, where $k=2$, $l=$ `Team_Name`, $l'=$ `Points_Delta` and $c(i,k)$ represents word span *took down*.

We can redefine the forward and backward terms as

$$\alpha(i,l,k) = \sum_{k^1,\ldots,k^{t-1}} \sum_{l^1,\ldots,l^{t-1}} \prod_{t=1}^{t(i)} \psi(i,k^t,l,l^{t-1})$$

$$\beta(i,l) = \sum_{k^t,\ldots,k^T} \sum_{l^{t+1},\ldots,l^T} \prod_{t=t(i)+1}^{T} \psi(i,k^t,l,l^{t-1}),$$

where the summations are taken over all possible segmentations and corresponding latent states.

We further define two terms:

$$\mu(i,l,k) = \sum_{\substack{k^1,\ldots,k^T\\k^{t(i)}=k}} \sum_{\substack{l^1,\ldots,l^T\\l^{t(i)}=l}} \prod_{t=1}^{t_I} \psi(i(t),k^t,l^t,l^{t-1})$$

$$\mu(i,l,l',k) = \sum_{\substack{k^1,\ldots,k^T\\k^{t(i)}=k}} \sum_{\substack{l^1,\ldots,l^T\\l^{t(i)}=l\\l^{t(i+k)}=l'}} \prod_{t=1}^{t_I} \psi(i(t),k^t,l^t,l^{t-1}).$$

Examples of these two μ's are shown in Figure 7 and Figure 8.

With this setting, the forward-backward algorithm can be implemented with the following definition of dynamic programming:

$$\alpha(i,l,k) = \sum_{l',k'} \alpha(i-k,l',k') \cdot \psi(i,k,l,l')$$
$$\beta(i,l) = \sum_{k',l'} \beta(i+k',l') \cdot \psi(i+k',k',l',l)$$
$$\mu(i,l,k) = \alpha(i,l,k) \cdot \beta(i,l)$$
$$\mu(i,k,l,l') = \sum_{k'} \alpha(i,l_i,k') \cdot \psi(i+k,k,l',l) \cdot \beta(i+k,l').$$

We can trivially obtain the soft counts of every transition and emission as before using μ.

C Parameter estimation

Due to the lack of supervision, we derive an expectation-maximization (EM) algorithm to estimate the parameters. Parameter estimation for multinomial distributions is the same as normal HMMs, so here we will only derive for the part used to model numerics-to-string correspondence, which resembles a Gaussian mixture model in format. To simplify our notation, we will temporarily neglect the semi-Markov scheme and other probabilistic models we adopt for other types of correspondences.

We rewrite the emission probability using the Bayes rule:

$$P_s(c|l) = P(c|l,v_l) = \frac{P(v_l|c,l)P(c|l)}{P(v_l|l)}$$

We have parameterized the emission and transi-

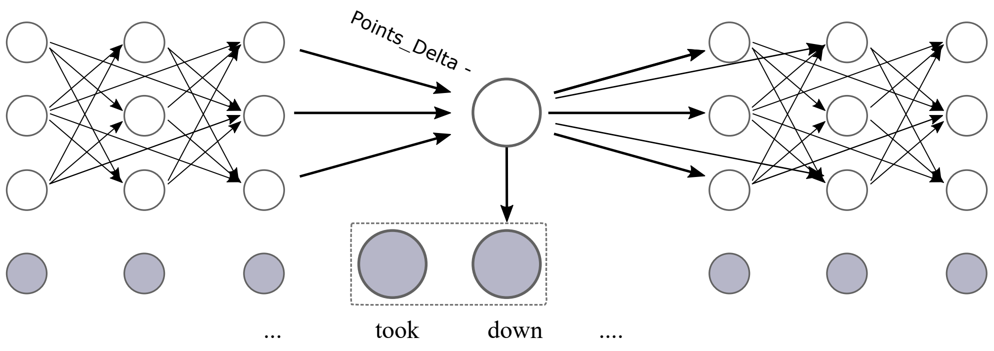


Figure 7: Example of $\mu(i,l,k)$ where $k=2$, $l=$ `Points_Delta` and $c(i,k)$ represents *took down*.

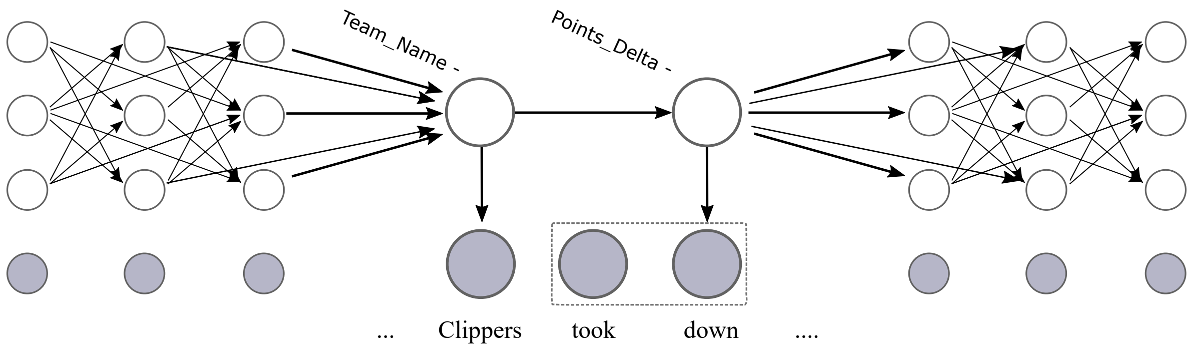


Figure 8: Example of $\mu(i,l,l',k)$ where $k=2$, $l=$ `Team_Name`, $l'=$ `Points_Delta`, and $c(i,k)$ represents *took down*.

tion probabilities as:

$$\begin{aligned}
P(v_l|c,l) &= \mathcal{N}(v_l;\mu_{c,l},\sigma_{c,l}),\\
P(v_l|l) &= \mathcal{N}(v_l;\mu_l,\sigma_l),\\
P(c|l) &= \eta_{c,l}, \quad \sum_c \eta_{c,l}=1,\\
P(l'|l) &= \xi_{l,l'}, \quad \sum_{l'} \xi_{l,l'}=1.
\end{aligned}$$

Let m be the number of possible labels in total. Set $\mathbf{z}^t$ as a one-hot vector of length m, with $z_l^t=1$ indicates the tag l is assigned to c^t at time t. (During the M-step in the EM process, this one-hot vector will be replaced with a soft count vector.) Similarly, we can parameterize the observable texts $\mathbf{x^t}$ as an n-dimensional vectors, where $x_c^t=1$ means word span c is observed at time t. We use v_l^t to denote the output value of label l at time t.

Then we use the following shorthand notation for the conditional log likelihood of complete data:

$$\log\mathcal{L}(\xi,\mu,\sigma,\eta)=\log\ P_s(\mathbf{l},\mathbf{c};\xi,\mu,\sigma,\eta),$$

where $\mathbf{l}$ denotes all tags annotated, and $\mathbf{c}$ denotes all word spans. Plug in all we obtained above:

$$\begin{aligned}
\log\mathcal{L}(\xi,\mu,\sigma,\eta) &= \sum_{l,l',t} z_l^t z_{l'}^{t+1}\log\xi_{l,l'}\\
&+\sum_{l,c,t} x_c^t z_c^t\big[\log\ \eta_{c,l}-\log\ \sigma_{c,l}+\log\ \sigma_l-\\
&\frac{(v_l^t-\mu_{c,l})^2}{2\sigma_{c,l}^2}+\frac{(v_l^t-\mu_l)^2}{2\sigma_l^2}\big].
\end{aligned}$$

Using the method of Lagrangian multiplier, we can find that parameter estimation could be implemented as the weighted mean and standard deviation:

$$\begin{aligned}
\hat{\xi}_{l,l'} &= \frac{\sum_t z_l^t z_{l'}^{t+1}}{\sum_t z_l^t}\\
\hat{\eta}_{c,l} &= \frac{\sum_t x_c^t z_l^t}{\sum_t z_l^t}\\
\hat{\mu}_{c,l} &= \frac{\sum_t x_c^t z_l^t v_l^t}{\sum_t x_c^t z_l^t}\\
\hat{\sigma}_{c,l} &= \sqrt{\frac{\sum_t x_c^t z_l^t (v_l^t-\hat{\mu}_{c,l})^2}{\sum_t x_c^t z_l^t}}
\end{aligned}$$

There is no need to estimate the parameters μ_l and σ_l, since we could treat them as a normalizer and marginalize them for every possible value v_l before inference.

D Implementation details for posterior regularization

Posterior regularization (PR) (Ganchev et al., 2010) is a mechanism to inject soft statistical constraints on the posterior distribution of the E-step in the EM algorithm. Formally, the objective function is changed from the original log likelihood $\mathcal{L}(\theta)$ into

$$\mathcal{J}(\theta, q) = \mathcal{L}(\theta) - KL(q(\mathbf{y}) \| p(\mathbf{y}|\mathbf{x}; \theta)),$$

where $\mathbf{x}$ and $\mathbf{y}$ denotes the observed and the latent variables, respectively, and q is restricted to be taken from the distributions which satisfy a few statistical constraints, i.e., $q \in \mathcal{Q} = \{q | \mathbb{E}_q[\mathbf{f}(\mathbf{x}, \mathbf{y})] \leq \mathbf{b}\}$.

We could implement our constraints as

$$\mathbb{E}[\mathbf{f}(\mathbf{x}, \mathbf{y})] = \sum_i \mathbb{E}[\mathbf{f}(\mathbf{x}, y_i)] \leq \mathbf{b_x},$$

where $\mathbf{f}$ are some features, and $\mathbf{b}$ is a list of boundaries for these features.

To tackle the garbage collection problem (Liang et al., 2009), we use a simple soft constraint for every sentence: At least r_0 portion of the words should be annotated with NULL, where $r_0 \in [0, 1]$. For HMMs-PR, we could write our feature function f and boundary b as:

$$\begin{aligned} f(\mathbf{w}, l_i) &= -\mathbb{1}(l_i = \texttt{NULL}), \\ b &= -r_0, \end{aligned}$$

where $\mathbf{w}$ are the individual words, and l_i is the tag annotated on the i-th word. Similarly, for Semi-HMMs-PR, we could write f and b as:

$$\begin{aligned} f(\mathbf{c}, l_i) &= -\mathbb{1}(l^t = \texttt{NULL}) \cdot k^t, \\ b &= -r_0, \end{aligned}$$

where $\mathbf{c}$ are the word spans, l^t is the tag annotated on the k-th word span, and k^t is the length of c^t.

With this simple soft constraint, most of the irrelevant tokens, such as the determiners and punctuations, could be correctly assigned with the NULL tag by the model.

We also tried to use PR to constrain the number of word span length of which is longer than 1, but it did not provide further improvement.

E Details of skipping null tags

Our model will sometimes annotate the NULL tag on irrelevant words, which could weaken the functionality of Markovian transitions. As a result, we would like to enable the transition probabilities between non-NULL tags only.

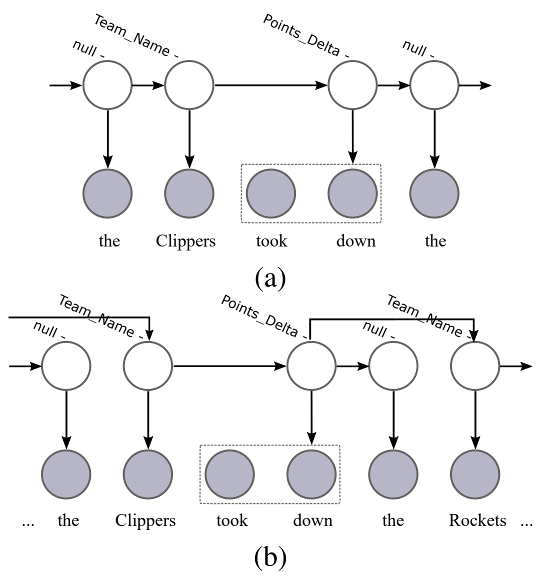


Figure 9: (a) Normal Semi-HMMs. (b) Semi-HMMs with skipping NULL scheme.

Take Figure 2 as an example. Before deploying the skipping NULL scheme, the Markov chain is

$$\texttt{Points\_Delta} \rightarrow \texttt{NULL} \rightarrow \texttt{Team\_Name},$$

where the transition probability from the label Points_Delta to the label Team_Name is intercepted by a meaningless tag NULL. To solve this problem, we replace $t(\texttt{Team\_Name}|\texttt{NULL})$ with $t(\texttt{Team\_Name}|\texttt{Points\_Delta})$, but all the emission probabilities remain unchanged, as demonstrated in Figure 9 (b).

To keep the Markovian property, we adopt the same method used in statistical machine translation (Brown et al., 1993). We duplicate m NULL tags, where m is the number of other tags. Let Tag-i be the i-th non-NULL tags, and NULL-i be the corresponding NULL tags, where $i = 1, \ldots, m$. The emission and transition probabilities are calculated through:

$$\begin{aligned} &e(c|\texttt{NULL-i}) = e(c|\texttt{NULL}) \\ &t(\texttt{NULL-i}|\texttt{Tag-j}) = \delta_{i,j} \\ &t(\texttt{AnyTag}|\texttt{NULL-i}) = t(\texttt{AnyTag}|\texttt{Tag-i}), \end{aligned}$$

where $\delta_{i,j}$ is the Kronecker delta taking the value 1 if $i = j$ and 0 otherwise, and AnyTag could represent any tag.

F More on error analysis and limitations

Since we use Gaussian distributions to model the probabilities from continuous output values to words, we might only find a soft boundary instead of an accurate hard boundary for lexical choices. For example, *double - double* is a term to describe an individual basketball performance in which a player accumulates a double-digit score in two categories. For most of the times, these two categories are rebounds and points, but apparently rebounds are more crucial. Thus our model successfully align word *double* to `Player_REB`. However, our model doesn't know that 10 is the threshold of double digit. In some generated texts, it might incorrectly state that a player has a *double-double* performance when he actually has only 9 rebounds. What makes it worse is that, since most of the players with such performance record less than 15 rebounds, the variance of Gaussian for tag `Player_REB` and word *double* is not large enough to cover values greater than 15.

Another severe problem derives from the limitation of our derived tag set. Despite of some irrelevant information described in the texts, many sentences need relatively more complicated algebraic operations or logic reasoning. For example, the reporter might say *in desperate need of a win*, which could be reasoned from the Win/Loss ratio. Another common situation is when the reporter talks about some inter-quarter information. *Hawks has a 10 - points lead before entering the fourth quarter* is an example which demands us to accumulate the first three quarters' scores and do a subtraction. This problem could be tackled by introducing more tags manually, but the more tags we adopt, the more difficult search space the model will encounter.

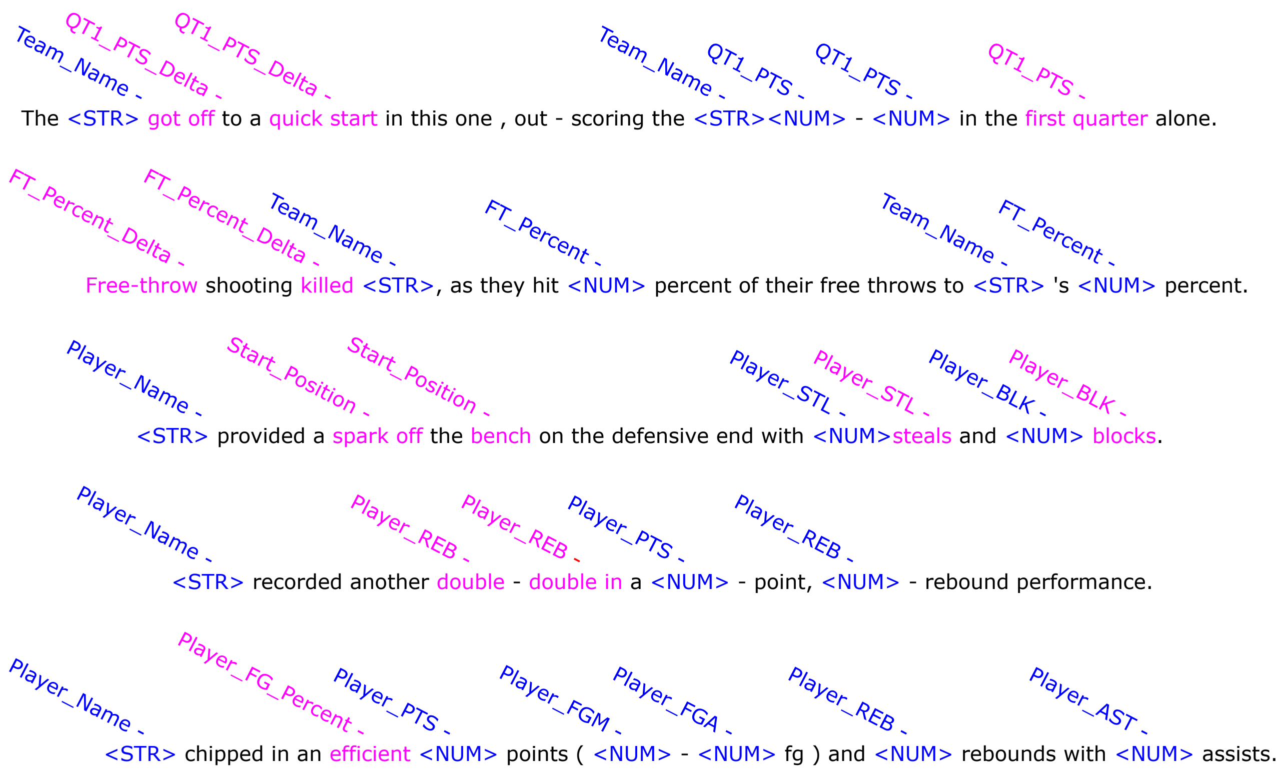
Compositional semantics is another problem with our models, which seems insolvable under current settings. For example, our model doesn't know that *defeated* and *was defeated* are completely opposite, since it could only derive some rules based on co-occurrences.

G Induced templates example

We list five samples of the induced templates in Figure 10. The slots are colored in blue, and the triggers are colored in magenta.

H Example of generation texts

We sampled one of our generated game summaries in Figure 11. The words in blue are originally slots in templates, and those in magenta are originally triggers.


The <STR> got off to a quick start in this one , out - scoring the <STR><NUM> - <NUM> in the first quarter alone.

Free-throw shooting killed <STR>, as they hit <NUM> percent of their free throws to <STR> 's <NUM> percent.

<STR> provided a spark off the bench on the defensive end with <NUM>steals and <NUM> blocks.

<STR> recorded another double - double in a <NUM> - point, <NUM> - rebound performance.

<STR> chipped in an efficient <NUM> points (<NUM> - <NUM> fg) and <NUM> rebounds with <NUM> assists.

Figure 10: Examples of the induced templates

The Los Angeles Clippers (14 - 5) blew out the New Orleans Pelicans (8 - 10) on Saturday , 120 - 100 . The Pelicans were killed in the assist - to - turnover ratio , with New Orleans committing 11 turnovers to 26 assists , while the Clippers handed out 34 assists to 7 turnovers . Los Angeles followed up a 34 point first quarter with a lowly 20 points in the second , which kept the Pelicans within striking distance after they had multiple opportunities to put the game away early . Anthony Davis led the team with 26 points , but grabbed just 3 boards . Point guard JJ Redick , whose name was frequently brought up in trade tumors , led the team with 21 points . Jamal Crawford actually led the team off the bench with 20 points , which he supplemented with 4 assist . Point guard Chris Paul also had a noteworthy 18 points and 16 assists . Matt Barnes provided 14 points , 4 rebounds and 1 steal . Tyreke Evans also pitched in with 13 points (5 - 13 fg , 2 - 4 ft) and 6 rebounds . Luke Babbitt pitched in with 12 points on a team - best 4 - for - 6 shooting night from the field . Omer Asik got the start at center and amassed 10 points , 10 rebounds and 1 steals . Rivers was able to shoot 3 - for - 8 from the field and 1 - for - 2 from the 3 - point line to score 8 points , while also grabbing 6 rebounds and handing out 6 assists . Jrue Holiday was a force in this game , going 3 - for - 7 from the field and 1 - for - 2 from the free throw line to score 7 points , while also adding 3 rebounds .

Figure 11: Example of generation