Supplementary Notes for 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

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

The quick brown fox jumps over a 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 1 shows an example. With the potential function defined above, the joint probability of the whole sentence can be written as

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

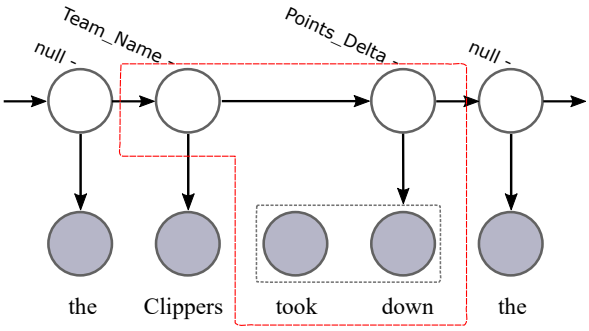

Figure 1: Example of a local potential function $\psi(i,k,l',l) = t(l'|l)\cdot e(c(i,k)|l')$, where $k=2$, $l=$ `Team_Name`, $l'=$ `Points_Delta` and $c(i,k)$ represents word span *took down*.

We can redefine the forward and backward terms as

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

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

We further define two terms:

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

Examples of these two μ's are shown in Figure 2 and Figure 3.

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.

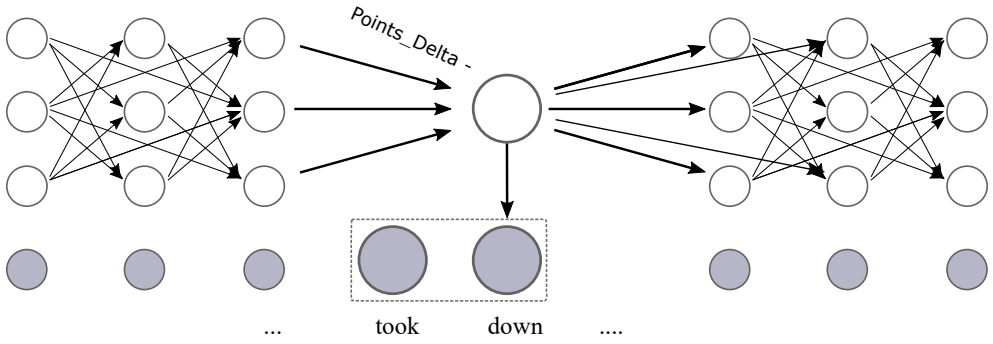

Figure 2: Example of $\mu(i, l, k)$ where $k = 2$, $l =$ `Points_Delta` and $c(i, k)$ represents *took down*.

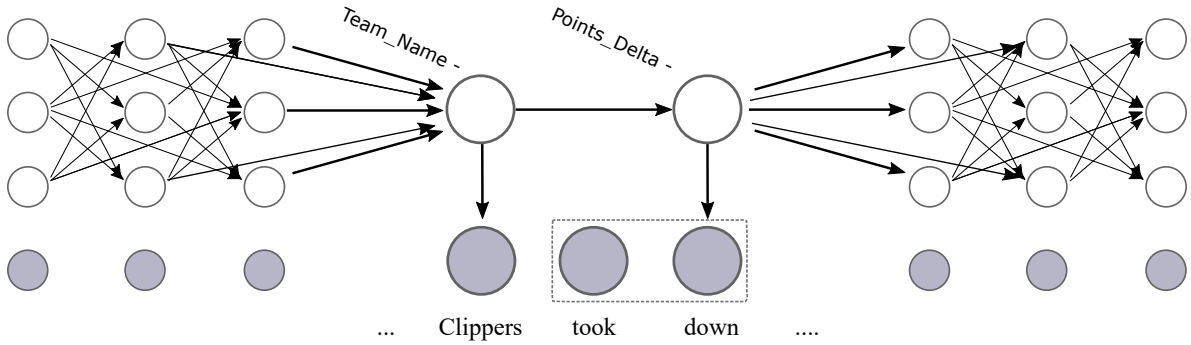

Figure 3: Example of $\mu(i, l, l', k)$ where $k = 2$, $l =$ `Team_Name`, $l' =$ `Points_Delta`, and $c(i, k)$ represents *took down*.

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 transition 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 imple-

mented as the weighted mean and standard deviation:

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

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

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

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:

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

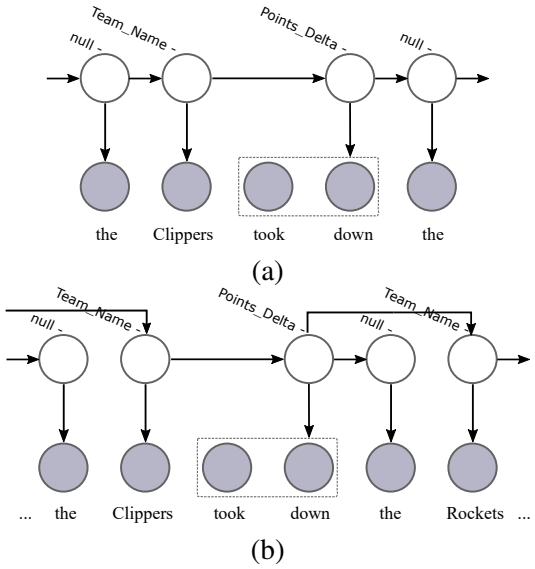

Figure 4: (a) Normal Semi-HMMs. (b) Semi-HMMs with skipping NULL scheme.

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:

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

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.

Take Figure 4 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 4 (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:

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

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 5. 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 6. The words in blue are originally slots in templates, and those in magenta are originally triggers.

References

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*, 19(2).

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

K. Ganchev, J. Gillenwater, and B. Taskar. 2010. Posterior Regularization for Structured Latent Variable Models. *Journal of Machine Learning Research (JMLR)*, 11.

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)*, August.

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

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

Figure 5: 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 6: Example of generation