VigDet: Knowledge Informed Neural Temporal Point Process for Coordination Detection on Social Media

Yizhou Zhang*, **Karishma Sharma***, **Yan Liu**
Department of Computer Science
Viterbi School of Engineering
University of Southern California
{zhangyiz,krsharma,yanliu.cs}@usc.edu

A Appendix

A.1 Proof of Theorem 1

Theorem 1. *Given a fixed inference of Q and a pre-defined $\phi_{\mathcal{G}}$, we have following inequality:*

$$\begin{aligned}\mathbb{E}_{Y\sim Q}\log P(Y|E,\mathcal{G}) &\geq \mathbb{E}_{Y\sim Q}\sum_{u\in\mathcal{V}}\log\frac{\exp\{\varphi_\theta(y_u,E_u)\}}{\sum_{1\leq m'\leq M}\exp\{\varphi_\theta(m',E_u)\}}+\textit{const}\\ &=\sum_{u\in\mathcal{V}}\sum_{1\leq m\leq M}Q_u(y_u=m)\log\frac{\exp\{\varphi_\theta(m,E_u)\}}{\sum_{1\leq m'\leq M}\exp\{\varphi_\theta(m',E_u)\}}+\textit{const}\end{aligned} \tag{1}$$

Proof. To simplify the notation, let us apply following notations:

$$\Phi_\theta(Y;E)=\sum_{u\in\mathcal{V}}\varphi_\theta(y_u,E_u),\qquad \Phi_{\mathcal{G}}(Y;\mathcal{G})=\sum_{(u,v)\in\mathcal{E}}\phi_{\mathcal{G}}(y_u,y_v,u,v) \tag{2}$$

Let us denote the set of all possible assignment as $\mathcal{Y}$, then we have:

$$\begin{aligned}\mathbb{E}_{y\sim Q}\log P(y|E,\mathcal{G}) &= \mathbb{E}_{Y\sim Q}\log\frac{\exp(\Phi(Y;E,\mathcal{G}))}{\sum_{Y'\in\mathcal{Y}}\exp(\Phi(Y';E,\mathcal{G}))}\\ &=\mathbb{E}_{y\sim Q}\Phi(Y;E,\mathcal{G})-\log\sum_{Y'\in\mathcal{Y}}\exp(\Phi(Y';E,\mathcal{G}))\\ &=\mathbb{E}_{y\sim Q}(\Phi_\theta(Y;E)+\Phi_{\mathcal{G}}(Y;\mathcal{G}))-\log\sum_{Y'\in\mathcal{Y}}\exp(\Phi(Y';E,\mathcal{G}))\end{aligned} \tag{3}$$

Because ϕ_G is pre-defined, $\Phi_{\mathcal{G}}(Y;\mathcal{G})$ is a constant. Thus, we have

$$\mathbb{E}_{y\sim Q}\log P(y|E,\mathcal{G})=\mathbb{E}_{y\sim Q}\Phi_\theta(Y;E)-\log\sum_{Y'\in\mathcal{Y}}\exp(\Phi(Y';E,\mathcal{G}))+\text{const} \tag{4}$$

Now, let us consider the $\log\sum_{Y'\in\mathcal{Y}}\exp(\Phi(Y';E,\mathcal{G}))$. Since ϕ_G is pre-defined, there must be an assignment Y_{max} that maximize $\Phi_{\mathcal{G}}(Y;\mathcal{G})$. Thus, we have:

$$\begin{aligned}\log\sum_{Y'\in\mathcal{Y}}\exp(\Phi(Y';E,\mathcal{G})) &\leq \log\sum_{Y'\in\mathcal{Y}}\exp(\Phi_\theta(Y;E)+\Phi_{\mathcal{G}}(Y_{\text{max}};\mathcal{G}))\\ &=\log\exp(\Phi_{\mathcal{G}}(Y_{\text{max}};\mathcal{G}))\sum_{Y'\in\mathcal{Y}}\exp(\Phi_\theta(Y;E))\\ &=\Phi_{\mathcal{G}}(Y_{\text{max}};\mathcal{G})+\log\sum_{Y'\in\mathcal{Y}}\exp(\Phi_\theta(Y;E))\end{aligned} \tag{5}$$

*Equally contributed

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

Since $\phi_{\mathcal{G}}$ is pre-defined, $\Phi_{\mathcal{G}}(Y_{\max};\mathcal{G}))$ is a constant during the optimization. Note that $\sum_{Y'\in\mathcal{Y}}\exp_\theta(\Phi(Y';E))$ sums up over all possible assignments $Y'\in\mathcal{Y}$. Thus, it is actually the expansion of following product:

$$\prod_{u\in\mathcal{V}}\sum_{1\le m'\le M}\exp(\varphi_\theta(m',E_u))=\sum_{Y'\in\mathcal{Y}}\prod_{u\in\mathcal{V}}\exp(\varphi_\theta(y'_u,E_u))=\sum_{Y'\in\mathcal{Y}}\exp(\Phi_\theta(Y';E)) \tag{6}$$

Therefore, for Q which is a mean-field distribution and φ_θ which model each account's assignment independently, we have:

$$\begin{aligned}
\mathbb{E}_{Y\sim Q}\log P(y|E,\mathcal{G}) &\ge \mathbb{E}_{y\sim Q}\Phi_\theta(Y;E)-\log\sum_{Y'\in\mathcal{Y}}\exp(\Phi_\theta(Y';E))+\text{const}\\
&=\mathbb{E}_{Y\sim Q}\Phi_\theta(Y;E)-\log\prod_{u\in\mathcal{V}}\sum_{1\le m'\le M}\exp(\varphi_\theta(m',E_u))+\text{const}\\
&=\mathbb{E}_{Y\sim Q}\Phi_\theta(Y;E)-\sum_{u\in\mathcal{V}}\log\sum_{1\le m'\le M}\exp(\varphi_\theta(m',E_u))+\text{const}\\
&=\mathbb{E}_{Y\sim Q}\sum_{u\in\mathcal{V}}\log\frac{\exp\{\varphi_\theta(y_u,E_u)\}}{\sum_{1\le m'\le M}\exp\{\varphi_\theta(m',E_u)\}}+\text{const}\\
&=\sum_{u\in\mathcal{V}}\sum_{1\le m\le M}Q_u(y_u=m)\log\frac{\exp\{\varphi_\theta(m,E_u)\}}{\sum_{1\le m'\le M}\exp\{\varphi_\theta(m',E_u)\}}+\text{const}
\end{aligned} \tag{7}$$

□

A.2 Detailed Justification to E-step

In the E-step, to acquire a mean field approximation $Q(Y)=\prod_{u\in\mathcal{V}}Q_u(y_u)$ that minimize the KL-divergence between Q and P, denoted as $D_{KL}(Q||P)$, we repeat following belief propagation operations until the Q converges:

$$Q_u(y_u=m)=\frac{\hat{Q}_u(y_u=m)}{Z_u}=\frac{1}{Z_u}\exp\{\varphi_\theta(m,E_u)+\sum_{v\in\mathcal{V}}\sum_{1\le m'\le M}\phi_{\mathcal{G}}(m,m',u,v)Q_v(y_v=m')\} \tag{8}$$

Here, we provide a detailed justification based on previous works [1, 2]. Let us recall the definition of the potential function $\Phi(Y;E,\mathcal{G})$ and the Gibbs distribution defined on it $P(Y|E,\mathcal{G})$:

$$\Phi(Y;E,\mathcal{G})=\sum_{u\in\mathcal{V}}\varphi_\theta(y_u,E_u)+\sum_{(u,v)\in\mathcal{E}}\phi_{\mathcal{G}}(y_u,y_v,u,v) \tag{9}$$

$$P(Y|E,\mathcal{G})=\frac{1}{Z}\exp(\Phi(Y;E,\mathcal{G})) \tag{10}$$

where $Z=\sum_Y\exp(\Phi(Y;E,\mathcal{G}))$. With above definitions, we have the following theorem:

Theorem 2. *(Theorem 11.2 in [1])*

$$D_{KL}(Q||P)=\log Z-\mathbb{E}_{Y\sim Q}\Phi(Y;E,\mathcal{G})-H(Q) \tag{11}$$

where $H(Q)$ is the information entropy of the distribution Q.

A more detailed derivation of the above equation can be found in the appendix of [2]. Since Z is fixed in the E-step, minimizing $D_{KL}(Q||P)$ is equivalent to maximizing $\mathbb{E}_{Y\sim Q}\Phi(Y;E,\mathcal{G})+H(Q)$. For this objective, we have following theorem:

Theorem 3. *(Theorem 11.9 in [1]) Q is a local maximum if and only if:*

$$Q_u(y_u=m)=\frac{1}{Z_u}\exp(\mathbb{E}_{Y-\{y_u\}\sim Q}\Phi(Y-\{y_u\};E,\mathcal{G}|y_u=m)) \tag{12}$$

where Z_u is the normalizer and $\mathbb{E}_{Y-\{y_u\}\sim Q}\Phi(Y-\{y_u\};E,\mathcal{G}|y_u=m)$ is the conditional expectation of Φ given that $y_u=m$ and the labels of other nodes are drawn from Q.

Meanwhile, note that the expectation of all terms in Φ that do not contain y_u is invariant to the value of y_u. Therefore, we can reduce all such terms from both numerator (the exponential function) and denominator (the normalizer Z_u) of Q_u. Thus, we have following corollary:

Corollary 1. *Q is a local maximum if and only if:*

$$Q_u(y_u = m) = \frac{1}{Z_u} \exp\{\varphi_\theta(m, E_u) + \sum_{v \in \mathcal{V}} \sum_{1 \le m' \le M} \phi_{\mathcal{G}}(m, m', u, v) Q_v(y_v = m')\} \tag{13}$$

where Z_u is the normalizer

A more detailed justification of the above corollary can be found in the explanation of Corollary 11.6 in the Sec 11.5.1.3 of [1]. Since the above local maximum is a fixed point of $D_{KL}(Q||P)$, fixed-point iteration can be applied to find such local maximum. More details such as the stationary of the fixed points can be found in the Chapter 11.5 of [1]

A.3 Details of Experiments

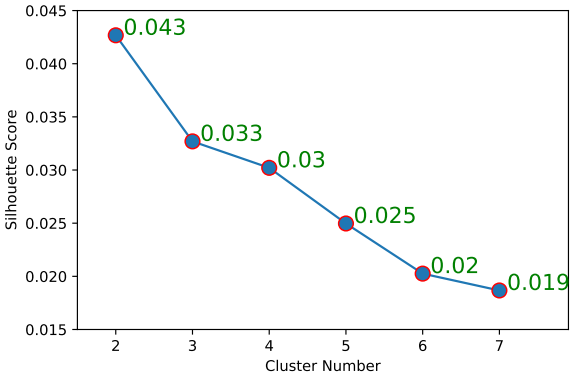

Figure 1: The silhouette scores of different group number.

A.3.1 Implementation details on IRA dataset

We split the sequence set to 75/15/15 fractions for training/validation/test sets. For the setting of AMDN and AMDN-HAGE [3] we use the default setting from the original paper including activity sequences of maximum length 128 (we split longer sequences), batch size of 256 (on 1 NVIDIA-2080Ti gpu), embedding dimension of 64, number of mixture components for the PDF in the AMDN part of 32, single head and single layer attention module, component number in the HAGE part of 2. Our implementation is totally based on PyTorch and Adam optimizer with 1e-3 learning rate and 1e-5 regularization (same as [3]). The number of loops in the EM algorithm is picked up from $\{1, 2, 3\}$ based on the performance on the validation account set. In each E-step, we repeat the belief propagation until convergence (within 10 iterations) to acquire the final inference. In each M-step, we train the model for max 50 epochs with early stopping based on validation objective function. The validation objective function is computed from the sequence likelihood on the 15% held-out validation sequences, and KL-divergence on the whole account set based on the inferred account embeddings in that iteration.

A.3.2 Implementation details on COVID-19 Vaccine Tweets dataset

We apply the Cubic Function based filtering because it shows better performance on unsupervised detection on IRA dataset. We follow all rest the settings of VigDet (CF) in IRA experiments except the GPU number (on 4 NVIDIA-2080Ti). Also, for this dataset, since we have no prior knowledge about how many groups exist, we first pre-train an AMDN by only maximizing its observed data likelihood on the dataset. Then we select the best cluster number that maximizes the silhouette score as the group number. The final group number we select is 2. The silhouette scores are shown in Fig.

Table 1: Results on unsupervised coordination detection (IRA) on Twitter in 2016 U.S. Election

Method	AP	AUC	F1	Prec	Rec	MaxF1	MacroF1
Co-activity	.169 ± .01	.525 ± .03	.246 ± .02	.178 ± .02	.407 ± .07	.271 ± .01	.495 ± .02
Clickstream	.165 ± .01	.532 ± .01	.21 ± .02	.206 ± .02	.216 ± .03	.21 ± .02	.531 ± .01
IRL	.239 ± .01	.687 ± .02	.353 ± .03	.275 ± .03	.494 ± .05	.386 ± .01	.588 ± .02
HP	.298 ± .03	.567 ± .03	.442 ± .03	.421 ± .02	.466 ± .04	.46 ± .03	.667 ± .01
A-H	.805 ± .03	.899 ± .02	.696 ± .05	.943 ± .03	.555 ± .06	.758 ± .03	.827 ± .03
A-H(Kmeans)	.82 ± .05	.933 ± .03	.73 ± .04	.909 ± .03	.612 ± .05	.77 ± .03	.845 ± .02
VigDet-PL(NF)	.816 ± .05	.933 ± .03	.73 ± .04	.852 ± .04	**.641 ± .06**	.765 ± .05	.844 ± .02
VigDet-E(NF)	.868 ± .03	**.955 ± .01**	.692 ± .07	**.964 ± .03**	.543 ± .07	.792 ± .04	.825 ± .04
VigDet(NF)	.856 ± .03	.951 ± .02	.698 ± .04	.958 ± .03	.551 ± .05	.788 ± .03	.828 ± .02
VigDet-PL(TL)	.833 ± .05	.94 ± .03	.707 ± .06	.896 ± .05	.59 ± .08	.778 ± .04	.832 ± .03
VigDet-E(TL)	.855 ± .03	.946 ± .03	.731 ± .03	.953 ± .03	.594 ± .04	**.796 ± .03**	.846 ± .02
VigDet(TL)	.861 ± .03	.946 ± .03	.734 ± .03	.951 ± .03	.599 ± .04	**.796 ± .03**	.848 ± .02
VigDet-PL(CF)	.845 ± .04	.95 ± .02	.719 ± .05	.914 ± .04	.596 ± .07	.793 ± .03	.839 ± .03
VigDet-E(CF)	.851 ± .04	.943 ± .03	.736 ± .03	.928 ± .03	.612 ± .04	.789 ± .03	.849 ± .02
VigDet(CF)	**.872 ± .03**	.95 ± .03	**.752 ± .03**	.917 ± .04	.639 ± .04	.793 ± .03	**.857 ± .02**

Table 2: Results on semi-supervised coordination detection (IRA) on Twitter in 2016 U.S. Election

Method	AP	AUC	F1	Prec	Rec	MaxF1	MacroF1
LPA(HP)	.633 ± .09	.768 ± .04	.681 ± .05	.762 ± .06	.618 ± .06	.716 ± .05	.815 ± .03
LPA(TL)	.697 ± .04	.859 ± .02	.623 ± .06	.885 ± .03	.486 ± .08	.661 ± .05	.786 ± .03
LPA(CF)	.711 ± .04	.853 ± .02	.608 ± .04	.665 ± .03	.564 ± .07	.683 ± .06	.772 ± .02
A-H + Semi-NN	.771 ± .04	.878 ± .03	.705 ± .04	.766 ± .06	.655 ± .04	.723 ± .04	.828 ± .02
A-H + GNN (HP)	.755 ± .06	.84 ± .05	.72 ± .07	.83 ± .14	.651 ± .08	.766 ± .05	.837 ± .04
A-H + GNN (CF)	.806 ± .06	.895 ± .04	.73 ± .07	.863 ± .06	.637 ± .09	.764 ± .06	.845 ± .04
A-H + GNN (TL)	.813 ± .05	.902 ± .03	.736 ± .06	.782 ± .06	.702 ± .09	.772 ± .06	.846 ± .03
VigDet-PL(NF)	.865 ± .03	.954 ± .01	.698 ± .06	.956 ± .03	.553 ± .07	.796 ± .04	.828 ± .03
VigDet-E(NF)	.868 ± .03	.955 ± .01	.692 ± .07	**.964 ± .03**	.543 ± .07	.792 ± .04	.825 ± .04
VigDet(NF)	.871 ± .03	.956 ± .01	.712 ± .06	.944 ± .04	.575 ± .07	.795 ± .04	.836 ± .03
VigDet-PL(TL)	.877 ± .04	.955 ± .01	.739 ± .08	.942 ± .04	.614 ± .09	.80 ± .06	.851 ± .04
VigDet-E(TL)	**.881 ± .04**	**.957 ± .01**	.734 ± .08	.946 ± .04	.604 ± .09	**.808 ± .05**	.848 ± .04
VigDet(TL)	.88 ± .04	**.957 ± .01**	.736 ± .08	.942 ± .04	.609 ± .09	**.808 ± .05**	.849 ± .04
VigDet-PL(CF)	.851 ± .03	.953 ± .01	.697 ± .06	.934 ± .03	.559 ± .07	.79 ± .04	.828 ± .03
VigDet-E(CF)	.871 ± .04	.952 ± .01	.744 ± .06	.928 ± .03	.624 ± .08	.797 ± .05	.853 ± .04
VigDet(CF)	.876 ± .03	.956 ± .01	**.761 ± .06**	.872 ± .07	**.681 ± .09**	.798 ± .05	**.862 ± .04**

1. After that, we train the VigDet on the dataset with group number of 2. As for the final threshold we select for detection, we set it as 0.8 because it maximizes the silhouette score on the final learnt embedding[2].

A.4 Detailed Performance

In Table. 1 and 2, we show detailed performance of our model and the baselines. Specifically, we provide the error bar of different methods. Also in the Sec. 4.1, we mention that we design two strategies to filter the edge weight because the naive edge weights suffer from group unbalance. Here, we give detailed results of applying naive edge weight without filtering in VigDet (denoted as VigDet (NF)). As we can see, compared with the version with filtering strategies, the recall scores of most variants with naive edge weight are significantly worse, leading to poor F1 score (excpet VigDet-PL(NF) in unsupervised setting, which performs significantly worse on threshold-free metrics like AP, AUC and MaxF1).

References

[1] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[2] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger,

[2] 0.9 can achieve better silhouette score but getting worse scores on some intermediate metrics like unreliable hyperlink source ratio

editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[3] Karishma Sharma, Yizhou Zhang, Emilio Ferrara, and Yan Liu. Identifying coordinated accounts on social media through hidden influence and group behaviours. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery Data Mining*, KDD '21, page 1441–1451, New York, NY, USA, 2021. Association for Computing Machinery.