Topic-Aware Neural Keyphrase Generation for Social Media Language

Yue Wang[1], Jing Li[2], Hou Pong Chan[1], Irwin King[1], Michael R. Lyu[1], Shuming Shi[2]
The Chinese University of Hong Kong[1] Tencent AI Lab[2]
[1]{yuewang, hpchan, king, lyu}@cse.cuhk.edu.hk, [2]{ameliajli, shumingshi}@tencent.com

Introduction

- **Keyphrase prediction:** distill salient information from massive posts
- **Challenges**:
  - Social media language is noisy and informal (data sparsity)
  - Prior work only extract keyphrases from the source post

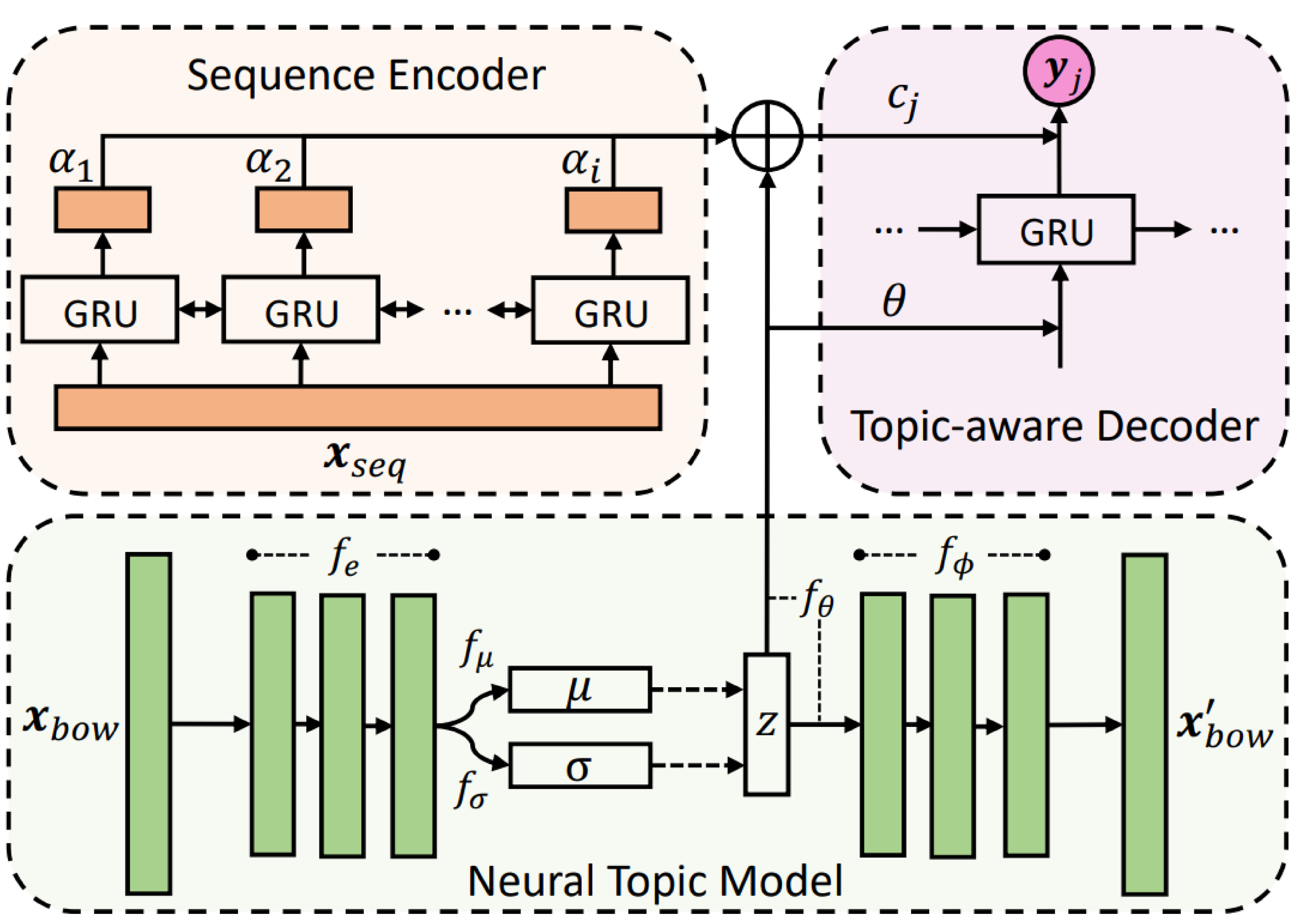
Source post with keyphrase "*super bowl*":
[S]: Somewhere, a wife that is not paying attention to the *game*, says "I want the *team* in *yellow pants* to *win*."

Relevant tweets:
$[T_1]$: I been a *steelers fan* way before *black* & *yellow* and this *super bowl*!
$[T_2]$: I will bet you the *team* with *yellow pants wins*.
$[T_3]$: Wiz Khalifa song '*black* and *yellow*" to spur the *pittsburgh steelers* and Lil Wayne is to sing "*green* and *yellow*' for the *packers*.

Our solution: topic-aware keyphrase generation model
- **Topic-aware**: post-level latent topics learned from corpus can alleviate the data sparsity
- **Sequence generation**: create new keyphrases

Our Approach

- **Overall framework**



Input — Post in word index x_{seq}; Post in bag of words x_{bow} → Keyphrase: $\langle y_1, y_2, \ldots, y_{|y|}\rangle$ — Output

- **Neural topic model (NTM)**

BoW Encoder	BoW Decoder
Prior latent variables ▪ $\mu = f_\mu(f_e(x_{bow}))$ ▪ $\log\sigma = f_\sigma(f_e(x_{bow}))$	▪ Draw latent variable $z \sim N(\mu, \sigma^2)$ ▪ Topic mixture $\theta = softmax(f_\theta(z))$ ▪ For each word w ∈ x: ▪ Draw word w ∼ $softmax(f_\varphi(\theta))$

- **Keyphrase generation (KG) model**
  - **Base model**: standard seq2seq with copy mechanism
  - **Advanced**: topic-aware sequence decoder

Topic θ from NTM →
- **Decoder state**: $s_j = f_{GRU}([u_j;\theta], s_{j-1})$
- **Attention**: $f_\alpha(\cdot) = v_\alpha^T tanh(W_\alpha[h_i; s_j; \theta] + b_\alpha)$
- **Copy switch**: $\lambda_j = \sigma(W_\lambda[u_j; s_j; c_j; \theta] + b_\lambda)$

- **Joint learning topics and keyphrases**

$$\left.\begin{aligned}
\mathcal{L}_{NTM} &= D_{KL}(p(\mathbf{z})\,||\,q(\mathbf{z}\,|\,\mathbf{x})) - \mathbb{E}_{q(\mathbf{z}\,|\,\mathbf{x})}[p(\mathbf{x}\,|\,\mathbf{z})],\\
\mathcal{L}_{KG} &= -\sum_{n=1}^{N}\log(Pr(\mathbf{y}_n\,|\,\mathbf{x}_n, \theta_n)),\\
\mathcal{L} &= \mathcal{L}_{NTM} + \gamma\cdot\mathcal{L}_{KG}
\end{aligned}\right\}$$

End-to-end training

Data Description

Source posts	# of posts	Avg len per post	# of KP per post	Source vocab
Twitter	44,113	19.52	1.13	34,010
Weibo	46,296	33.07	1.06	98,310
StackExchange	49,447	87.94	2.43	99,775
Target KP	**|KP|**	**Avg len per KP**	**% of abs KP**	**Target vocab**
Twitter	4,347	1.92	71.35	4,171
Weibo	2,136	2.55	75.74	2,833
StackExchange	12,114	1.41	54.32	10,852

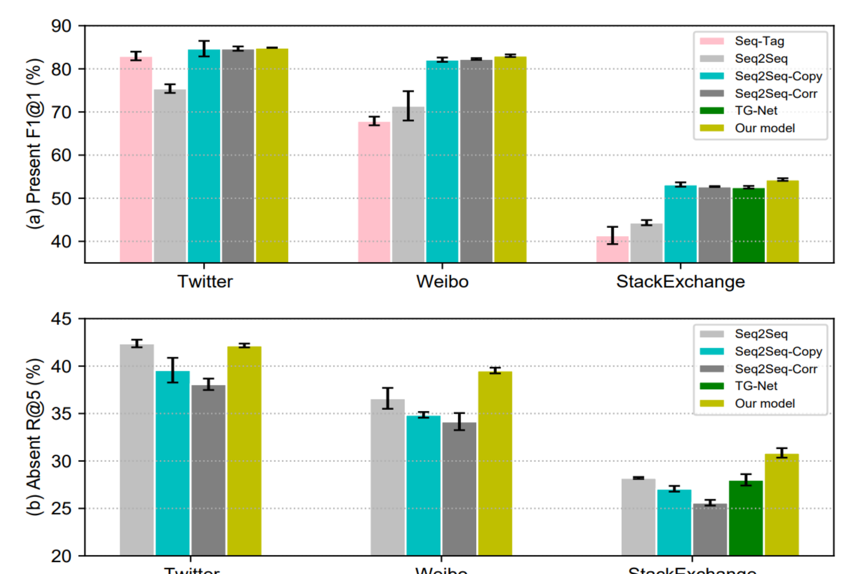
- 80% training
- 10% validation
- 10% test

High absent rate

Experiment Results

- **Main results**

Model	Twitter F1@1	Twitter F1@3	Twitter MAP	Weibo F1@1	Weibo F1@3	Weibo MAP	StackExchange F1@3	StackExchange F1@5	StackExchange MAP
Baselines									
MAJORITY	9.36	11.85	15.22	4.16	3.31	5.47	1.79	1.89	1.59
TF-IDF	1.16	1.14	1.89	1.90	1.51	2.46	13.50	12.74	12.61
TEXTRANK	1.73	1.94	1.89	0.18	0.49	0.57	6.03	8.28	4.76
KEA	0.50	0.56	0.50	0.20	0.20	0.20	15.80	15.23	14.25
State of the arts									
SEQ-TAG	22.79±0.3	12.27±0.2	22.44±0.3	16.34±0.2	8.99±0.1	16.53±0.3	17.58±1.6	12.82±1.2	19.03±1.3
SEQ2SEQ	34.10±0.5	26.01±0.3	41.11±0.3	28.17±1.7	20.59±0.9	34.19±1.7	22.99±0.3	20.65±0.2	23.95±0.3
SEQ2SEQ-COPY	36.60±1.1	26.79±0.5	43.12±1.2	32.01±0.3	22.69±0.2	38.01±0.1	31.53±0.1	27.41±0.2	33.45±0.1
SEQ2SEQ-CORR	34.97±0.8	26.13±0.4	41.64±0.5	31.64±0.7	22.24±0.5	37.47±0.8	30.89±0.3	26.97±0.2	32.87±0.6
TG-NET	-	-	-	-	-	-	32.02±0.3	27.84±0.3	34.05±0.4
Our model	**38.49**±0.3	**27.84**±0.0	**45.12**±0.2	**34.99**±0.3	**24.42**±0.2	**41.29**±0.4	**33.41**±0.2	**29.16**±0.1	**35.52**±0.1



Present vs. Absent keyphrase

- Social media keyphrase prediction is challenging
- Seq2seq-based keyphrase generation models are effective
- Latent topics are consistently helpful for indicating keyphrases, especially for absent keyphrases

- **Topic modeling**

Datasets	Twitter	StackExchange
LDA	41.12	35.13
BTM	43.12	43.52
NTM	43.82	43.04
Our model	**46.28**	**45.12**

(a) Topic coherence (C_V scores)

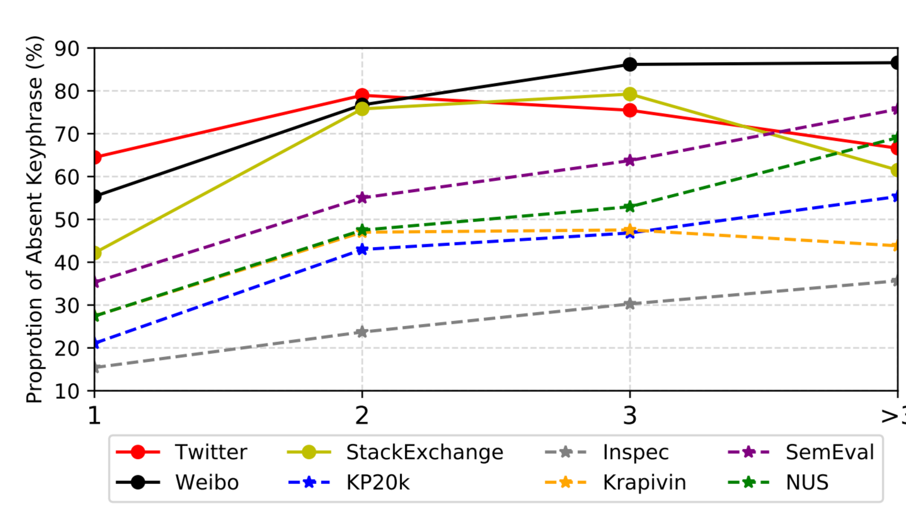
LDA	bowl super quote steeler jan watching egypt playing glee girl
BTM	bowl super anthem national christina aguilera fail word brand playing
NTM	super bowl eye protester winning watch halftime ship sport mena
Our model	bowl super yellow green packer steeler nom commercial win winner

(b) Sample topics for "*super bowl*"

- **Further discussions**

Model	Twitter	Weibo	SE
SEQ2SEQ-COPY	36.60	32.01	31.53
Our model (*separate train*)	36.75	32.75	31.78
Our model (*w/o topic-attn*)	37.24	32.42	32.34
Our model (*w/o topic-state*)	37.44	33.48	31.98
Our full model	**38.49**	**34.99**	**33.41**

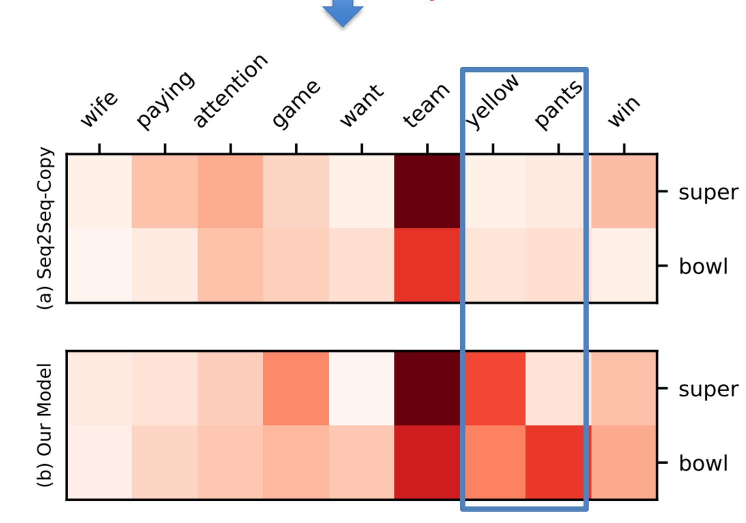
(a) Ablation study



(c) KP absent rate across other text genres

For tweet S, our model correctly predicts *"super bowl"*, while the seq2seq-copy model without topic guidance wrongly predicts *"team follow back"*

Why? Visualize attention!



1^{st} Topic	steeler national team packer win

(b) Case study

Conclusion & Future Work

- We are the first to propose a topic-aware keyphrase generation model that allows end-to-end training with latent topics
- We newly construct three social media datasets for this task
- Extensive experiments demonstrate the effectiveness of our proposed model for social media language

- Explore how to explicitly leverage the topic-word information
- Extend to other text generation tasks

Find our code & data