Joint Many-Task Model: NN for multiple NLP tasks

Statistical Methods for Natural Language Semantics

Steven van de Graaf, Azamat Omuraliev

Plan of the presentation

- (Brief) intro to multi-task learning
- Model architecture
- Model training
- Results
- Closing thoughts

JMT models?

- Solving multiple learning tasks at the same time →

  → exploiting commonalities and differences across tasks
- Improve performance through:
  - Data amplification
  - Representation bias
  - Attribute selection
  - Eavesdropping

Why this paper?

- **POS + Chunking + Dependency + Relatedness + Entailment**
- Hard parameter sharing + Layer per task
- Word embeddings + character embeddings
- Label embeddings

Why this paper?

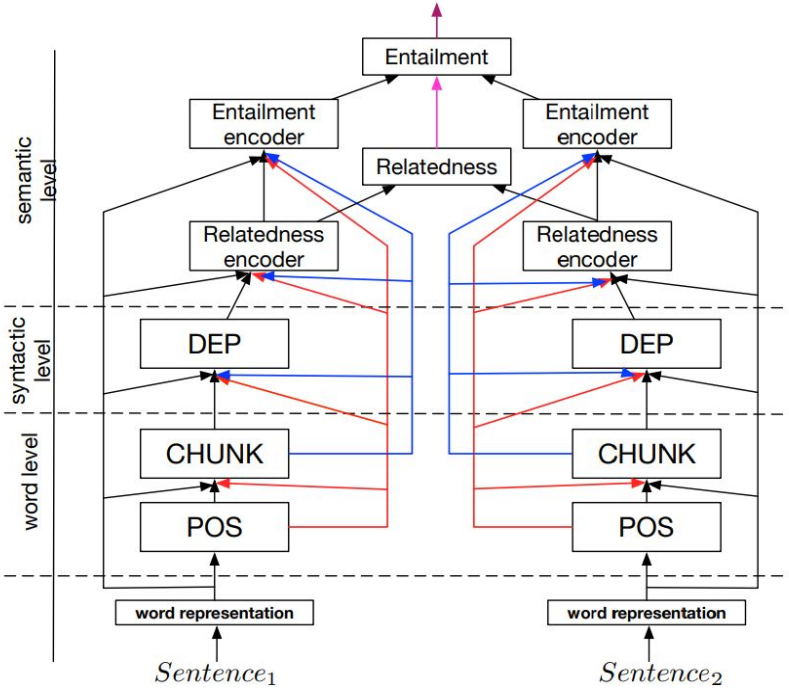
- POS + Chunking + Dependency + Relatedness + Entailment
- **Hard parameter sharing** + **Layer per task**
  - Regularization (to avoid forgetting)
  - Hierarchical order of layers
- Word embeddings + character embeddings
- Label embeddings

Why this paper?

- POS + Chunking + Dependency + Relatedness + Entailment
- Hard parameter sharing + Layer per task
- **Word embeddings + character embeddings**
  - Shortcut connections
- Label embeddings

Architecture of the JMT model

Joint Many-Task architecture



Predict increasingly complex NLP tasks at successively deeper layers

	Word and char	Only word
Embedding	leaning	stood
	kneeling	stands
	saluting	sit
	clinging	pillar
	railing	cross-legged
POS	warning	ladder
	waxing	rc6280
	dunking	bethle
	proving	warning
	tipping	f-a-18
Chunking	applauding	fight
	disdaining	favor
	pickin	pick
	readjusting	rejoin
	reclaiming	answer
Dependency	guaranteeing	patiently
	resting	hugging
	grounding	anxiously
	hanging	resting
	hugging	disappointment
Relatedness	stood	stood
	stands	unchallenged
	unchallenged	stands
	notwithstanding	beside
	judging	exists
Entailment	nudging	beside
	skirting	stands
	straddling	pillar
	contesting	swung
	footing	ovation

Word and character representations

Word: skip-gram

Character: n-gram skip-gram

→ concatenated

example for “standing”

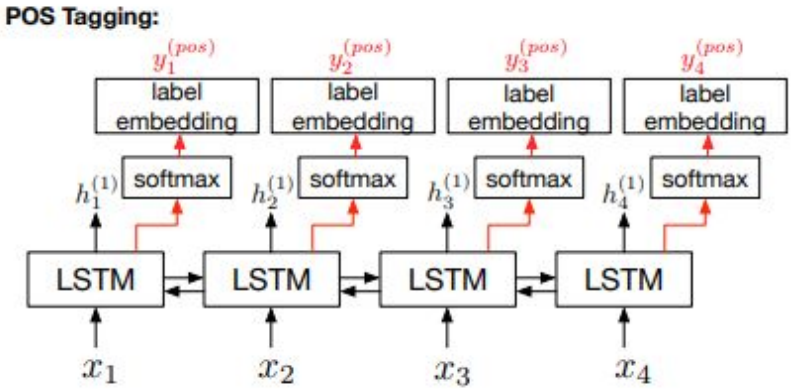
POS tagging

Input:

word embedding

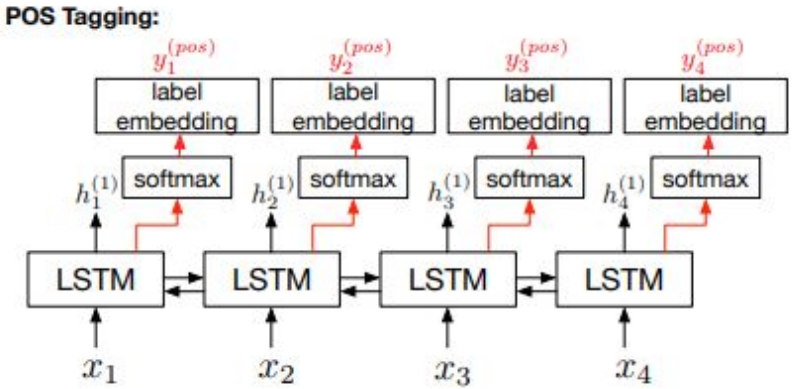
Output:

label embedding



POS tagging + Chunking

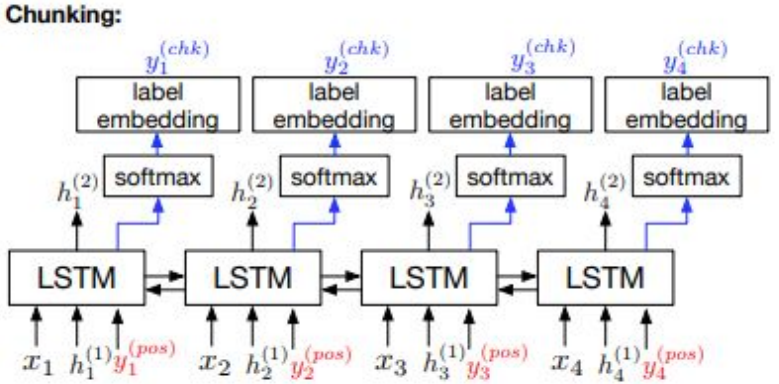
Input:
word embedding
Output:
label embedding



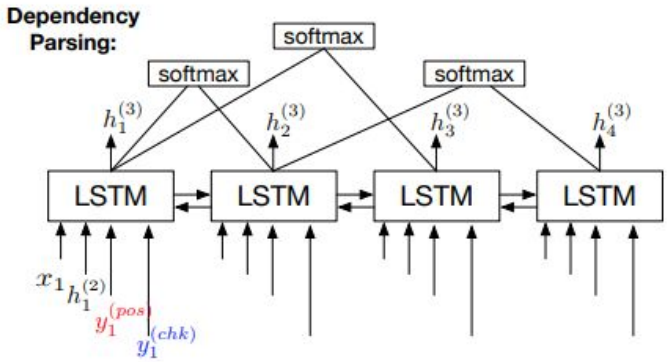
Input:
word embedding + POS embedding
+ POS hidden state

$$h_t = [\overrightarrow{h}_t; \overleftarrow{h}_t]$$

Output:
label embedding



Dependency parsing



Instead of building dependency trees, predict parent node for each word.

$$p(j|h_t^{(3)}) = \frac{\exp\left(m\left(t,j\right)\right)}{\sum_{k=1,k\neq t}^{L+1}\exp\left(m\left(t,k\right)\right)}$$

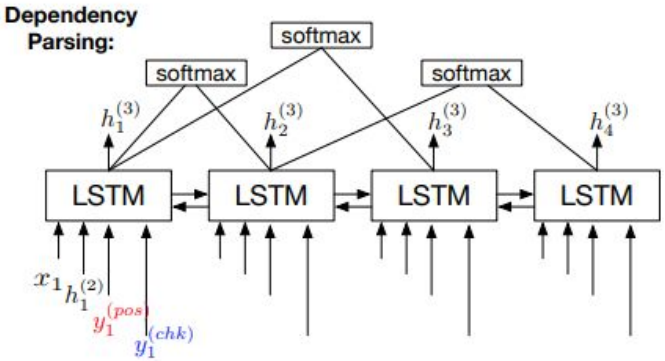
Note: double set of parameters

LSTM weights, DEP matching function weights

$$\theta_{\text{dep}} = (W_{\text{dep}}, b_{\text{dep}}, W_d, r, E_{\text{POS}}, E_{\text{chk}}, \theta_e)$$

Zhang, X., Cheng, J., & Lapata, M. (2016). Dependency parsing as head selection. *arXiv preprint arXiv:1606.01280*.

Dependency parsing



Input:

word embedding + POS embedding +

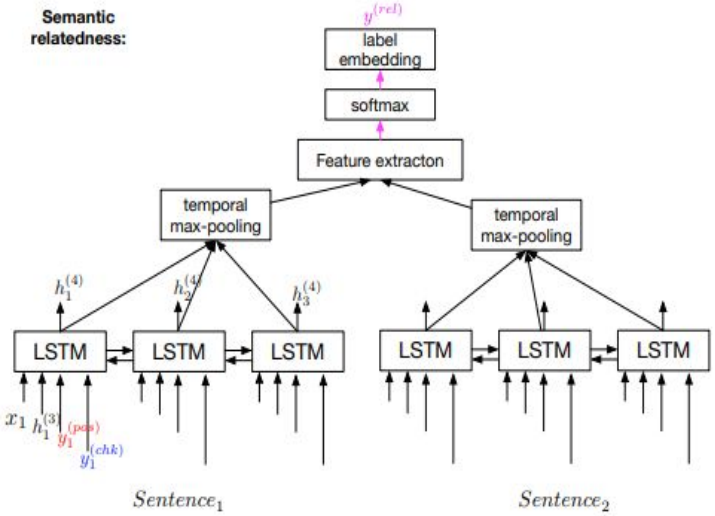
CHUNK embedding +

CHUNK hidden state

Output:

Label (greedy selection)

Relatedness + Entailment



Input:

word embedding + POS embedding +

CHUNK embedding +

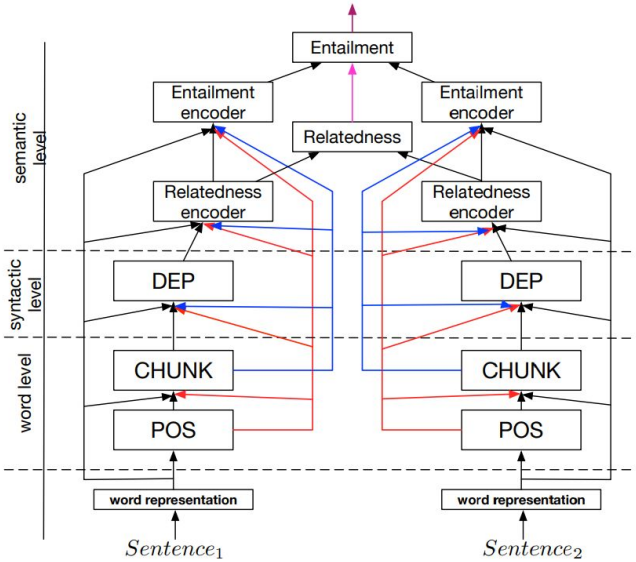
DEP hidden state

Max-pooling: $h_{\mathbf{s}}^{(4)} = \max\left(h_1^{(4)}, h_2^{(4)}, \ldots, h_L^{(4)}\right)$

Feature vector: $d_1(s, s') = \left[\left|h_{\mathbf{s}}^{(4)} - h_{\mathbf{s}'}^{(4)}\right|; h_{\mathbf{s}}^{(4)} \odot h_{\mathbf{s}'}^{(4)}\right]$

Training the JMT model

Overview



Trained **jointly** over **all** datasets in **full**, in **order** of the tasks (complexity):

1. POS (POS tagging)
2. CHK (Chunking)
3. DEP (Dependency parsing)
4. REL (Semantic relatedness)
5. ENT (Textual Entailment)

Pre-training

Word embeddings are pre-trained using Skip-Gram (SG) with negative sampling

Similarly, character *n*-gram embeddings are trained using SG

These are finetuned further during model training

POS tagging

Set of model parameters: $\theta_{POS} = (W_{POS}, b_{POS}, \theta_e)$

Objective function:

Task objective

$$J_1(\theta_{\text{POS}}) = -\sum_s \sum_t \log p(y_t^{(1)} = \alpha | h_t^{(1)}) + \lambda \|W_{\text{POS}}\|^2 + \delta \|\theta_e - \theta'_e\|^2, \quad (6)$$

L2-norm regularization (task-specific weight-decay)

Succesive regularization

POS tagging

Task objective

$$-\sum_{s}\sum_{t}\log p(y_t^{(1)} = \alpha | h_t^{(1)})$$

The probability that the correct label (α) is assigned to w_t of sentence *s*

$$p(y_t^{(1)} = \alpha | h_t^{(1)})$$

POS tagging

L2-norm regularization (task-specific weight-decay)

$$\lambda \|W_{\text{POS}}\|^2$$

λ is a hyperparameter

POS tagging

Successive regularization $\delta \|\theta_e - \theta'_e\|^2$

- δ is a hyperparameter
- θ_e are the embedding parameters of the current epoch
- θ'_e are the embedding parameters after training of the last task on the previous epoch

Avoids forgetting information learned previously (the embedding parameters of the current epoch shouldn't deviate too much from those of the previous epoch)

Chunking

Set of model parameters: $\theta_{\text{chk}} = (W_{\text{chk}}, b_{\text{chk}}, E_{\text{pos}}, \theta_e)$

Objective function:

$$J_2(\theta_{\text{chk}}) = -\sum_s \sum_t \log p(y_t^{(2)} = \alpha | h_t^{(2)}) + \lambda \|W_{\text{chk}}\|^2 + \delta \|\theta_{\text{POS}} - \theta'_{\text{POS}}\|^2, \quad (7)$$

Dependency parsing

Set of model parameters:

$$\theta_{\text{dep}} = (W_{\text{dep}}, b_{\text{dep}}, W_d, r, E_{\text{POS}}, E_{\text{chk}}, \theta_e)$$

Objective function:

$$J_3(\theta_{\text{dep}}) = -\sum_s \sum_t \log p(\alpha|h_t^{(3)}) p(\beta|h_t^{(3)}, h_\alpha^{(3)}) + \lambda(\|W_{\text{dep}}\|^2 + \|W_d\|^2) + \delta\|\theta_{\text{chk}} - \theta'_{\text{chk}}\|^2, \quad (8)$$

Semantic Relatedness

Set of model parameters: $\theta_{\text{rel}} = (W_{\text{rel}}, b_{\text{rel}}, E_{\text{POS}}, E_{\text{chk}}, \theta_e)$

Objective function:

$$J_4(\theta_{\text{rel}}) = \sum_{(s,s')} \text{KL}\left(\hat{p}(s,s') \middle\| p(h_s^{(4)}, h_{s'}^{(4)})\right) + \lambda \|W_{\text{rel}}\|^2 + \delta \|\theta_{\text{dep}} - \theta'_{\text{dep}}\|^2, \quad (9)$$

Semantic Relatedness

$$\mathrm{KL}\left(\hat{p}(s, s') \middle\| p(h_s^{(4)}, h_{s'}^{(4)})\right)$$

KL-divergence between:

- the ***true*** **distribution** over the relatedness scores
- the ***predicted*** **distribution** over the relatedness scores

Textual Entailment

Set of model parameters:

$$\theta_{\text{ent}} = (W_{\text{ent}}, b_{\text{ent}}, E_{\text{POS}}, E_{\text{chk}}, E_{\text{rel}}, \theta_e)$$

Objective function:

$$J_5(\theta_{\text{ent}}) = -\sum_{(s,s')} \log p(y^{(5)}_{(s,s')} = \alpha | h^{(5)}_s, h^{(5)}_{s'}) + \lambda \|W_{\text{ent}}\|^2 + \delta \|\theta_{\text{rel}} - \theta'_{\text{rel}}\|^2, \quad (10)$$

Experiments & Results

Experimental settings

Task	Dataset	Metric
POS	Wall Street Journal (WSJ) portion of the Penn Treebank	Word-level accuracy
CHK	WSJ	F-measure
DEP	WSJ	Unlabeled Attachment Score (UAS) and Labeled Attachment Score (LAS)
REL	SICK dataset	MSE
ENT	SICK dataset	Accuracy

JMT model (variant) performance on the test sets

		Single	JMT_{all}	JMT_{AB}	JMT_{ABC}	JMT_{DE}	JMT_{CD}	JMT_{CE}
A ↑	POS	97.45	97.55	97.52	97.54	n/a	n/a	n/a
B ↑	Chunking	95.02	n/a	95.77	n/a	n/a	n/a	n/a
C ↑	Dependency UAS	93.35	94.67	n/a	94.71	n/a	93.53	93.57
	Dependency LAS	91.42	92.90	n/a	92.92	n/a	91.62	91.69
D ↓	Relatedness	0.247	0.233	n/a	n/a	0.238	0.251	n/a
E ↑	Entailment	81.8	86.2	n/a	n/a	86.8	n/a	82.4

Table 1: Test set results for the five tasks. In the relatedness task, the lower scores are better.

All results of the 5 tasks are improved upon with the JMT model

The model variants show that the JMT model improves both the high-level and low-level tasks

Comparison with published results: POS & CHK

Method	Acc. ↑
JMT_{all}	97.55
Ling et al. (2015)	**97.78**
Kumar et al. (2016)	97.56
Ma and Hovy (2016)	97.55
Søgaard (2011)	97.50
Collobert et al. (2011)	97.29
Tsuruoka et al. (2011)	97.28
Toutanova et al. (2003)	97.27

Table 2: POS tagging results.

Method	F1 ↑
JMT_{AB}	**95.77**
Single	95.02
Søgaard and Goldberg (2016)	95.56
Suzuki and Isozaki (2008)	95.15
Collobert et al. (2011)	94.32
Kudo and Matsumoto (2001)	93.91
Tsuruoka et al. (2011)	93.81

Table 3: Chunking results.

Comparison with published results: DEP, REL & ENT

Method	UAS ↑	LAS ↑
JMT_{all}	94.67	92.90
Single	93.35	91.42
Dozat and Manning (2017)	**95.74**	**94.08**
Andor et al. (2016)	94.61	92.79
Alberti et al. (2015)	94.23	92.36
Zhang et al. (2017)	94.10	91.90
Weiss et al. (2015)	93.99	92.05
Dyer et al. (2015)	93.10	90.90
Bohnet (2010)	92.88	90.71

Table 4: Dependency results.

Method	MSE ↓
JMT_{all}	**0.233**
JMT_{DE}	0.238
Zhou et al. (2016)	0.243
Tai et al. (2015)	0.253

Table 5: Semantic relatedness results.

Method	Acc. ↑
JMT_{all}	86.2
JMT_{DE}	**86.8**
Yin et al. (2016)	86.2
Lai and Hockenmaier (2014)	84.6

Table 6: Textual entailment results.

Analysis of model architectures

Takeaways:

- The “Shortcut” connections (SC) and the output label embeddings (LE) of the previous layers are important for model performance
- Having different layers for different tasks performs best
- *Successive* regularization mostly affects the chunking task (small dataset)
- Using the bi-LSTM hidden states of the previous layer (task) works better than just “stacking” the various bi-LSTM layers

Analysis of model architectures

Takeaways:

- The order of tasks during training is important for model performance
- Joint learning is more important than making the models deeper only for single tasks
- Using the *n*-gram character embeddings next to the word embeddings is helpful in improving model performance

Closing thoughts

Likes

- It introduces a powerful, novel model architecture for JTL problems
- It addresses ways of dealing with the issues of forgetting / interference
- Elaborate analysis of the model architectures

Dislikes

- Often-times quite confusing
- No statistical significances
- Lacks results on the training process and dynamics of model performance
- Little investigation into how the model benefits from the MTL setup

Future research

- Exploring other training strategies (model convergence)
- Exploring using more tasks
- Exploring a different layer ordering (reversed, maybe?)
- Incorporate character-based embeddings into the JMT model
- Incorporating an attention mechanism to the dependency parsing
- Using the output of the dependency layer in further layers

Related work

POS + CHUNK + LM [1]

POS + CHUNK + CCG [2]

POS + DEP [3]

Entity detection + relation extraction [4]

[1] Godwin, J., Stenetorp, P., & Riedel, S. (2016). Deep semi-supervised learning with linguistically motivated sequence labeling task hierarchies. *arXiv preprint arXiv:1612.09113*.

[2] Søgaard, A., & Goldberg, Y. (2016). Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (Vol. 2, pp. 231-235).

[3] Zhang, Y., & Weiss, D. (2016). Stack-propagation: Improved representation learning for syntax. *arXiv preprint arXiv:1603.06598*.

[4] Miwa, M., & Bansal, M. (2016). End-to-end relation extraction using lstms on sequences and tree structures. arXiv preprint arXiv:1601.00770.

Questions and/or comments?

Appendix

Results: Importance of “Shortcut” connections and label embeddings

	JMT_{all}	w/o SC	w/o LE	w/o SC&LE
POS	97.88	97.79	97.85	97.87
Chunking	97.59	97.08	97.40	97.33
Dependency UAS	94.51	94.52	94.09	94.04
Dependency LAS	92.60	92.62	92.14	92.03
Relatedness	0.236	0.698	0.261	0.765
Entailment	84.6	75.0	81.6	71.2

Table 7: Effectiveness of the Shortcut Connections (SC) and the Label Embeddings (LE).

Takeaway:

- The “Shortcut” connections (SC) and the output label embeddings (LE) of the previous layers are important for model performance

Results: Importance of different layers for different tasks

	JMT_{ABC}	w/o SC&LE	All-3
POS	97.90	97.87	97.62
Chunking	97.80	97.41	96.52
Dependency UAS	94.52	94.13	93.59
Dependency LAS	92.61	92.16	91.47

Table 8: Effectiveness of using different layers for different tasks.

"All-3" shows the results of using the "highest" (ie the 3rd layer) for all 3 tasks

Takeaway:

- Having different layers for different tasks performs best, also when the number of model parameters are equal

Results: Importance of *successive* regularization & vertical connections

	JMT_{all}	w/o SR	w/o VC
POS	97.88	97.85	97.82
Chunking	97.59	97.13	97.45
Dependency UAS	94.51	94.46	94.38
Dependency LAS	92.60	92.57	92.48
Relatedness	0.236	0.239	0.241
Entailment	84.6	84.2	84.8

Table 9: Effectiveness of the Successive Regularization (SR) and the Vertical Connections (VC).

Takeaways:

- *Successive* regularization mostly affects the chunking task
- Using the bi-LSTM hidden states of the previous layer (task) works better than just “stacking” the various bi-LSTM layers

Results: Importance of layer ordering

	JMT_{all}	Random
POS	97.88	97.83
Chunking	97.59	97.71
Dependency UAS	94.51	94.66
Dependency LAS	92.60	92.80
Relatedness	0.236	0.298
Entailment	84.6	83.2

Table 10: Effects of the order of training.

Takeaway:

- The order of tasks during training is important for model performance!

Results: Importance of depth

	Single	Single+
POS	97.52	
Chunking	95.65	96.08
Dependency UAS	93.38	93.88
Dependency LAS	91.37	91.83
Relatedness	0.239	0.665
Entailment	83.8	66.4

Table 11: Effects of depth for the *single* tasks.

Takeaways:

- Deeper is not always better
- Joint learning is more important than making the models complex only for single tasks

Results: Importance of *n*-gram character embeddings

Single	W&C	Only W
POS	97.52	96.26
Chunking	95.65	94.92
Dependency UAS	93.38	92.90
Dependency LAS	91.37	90.44

Table 12: Effects of the character embeddings.

Takeaways:

- Using the *n*-gram character embeddings next to the word embeddings is helpful in improving model performance!

UNIVERSITY OF AMSTERDAM
Faculty of Science

Detect Rumor and Stance Jointly by Neural Multi-task Learning

authors: Jing Ma, Wei Gao & Kam-Fai Wong

Presented by: Freddy de Greef

Rumor detection

LOTTERY WINNER ARRESTED FOR DUMPING $200,000 OF MANURE ON EX-BOSS' LAWN

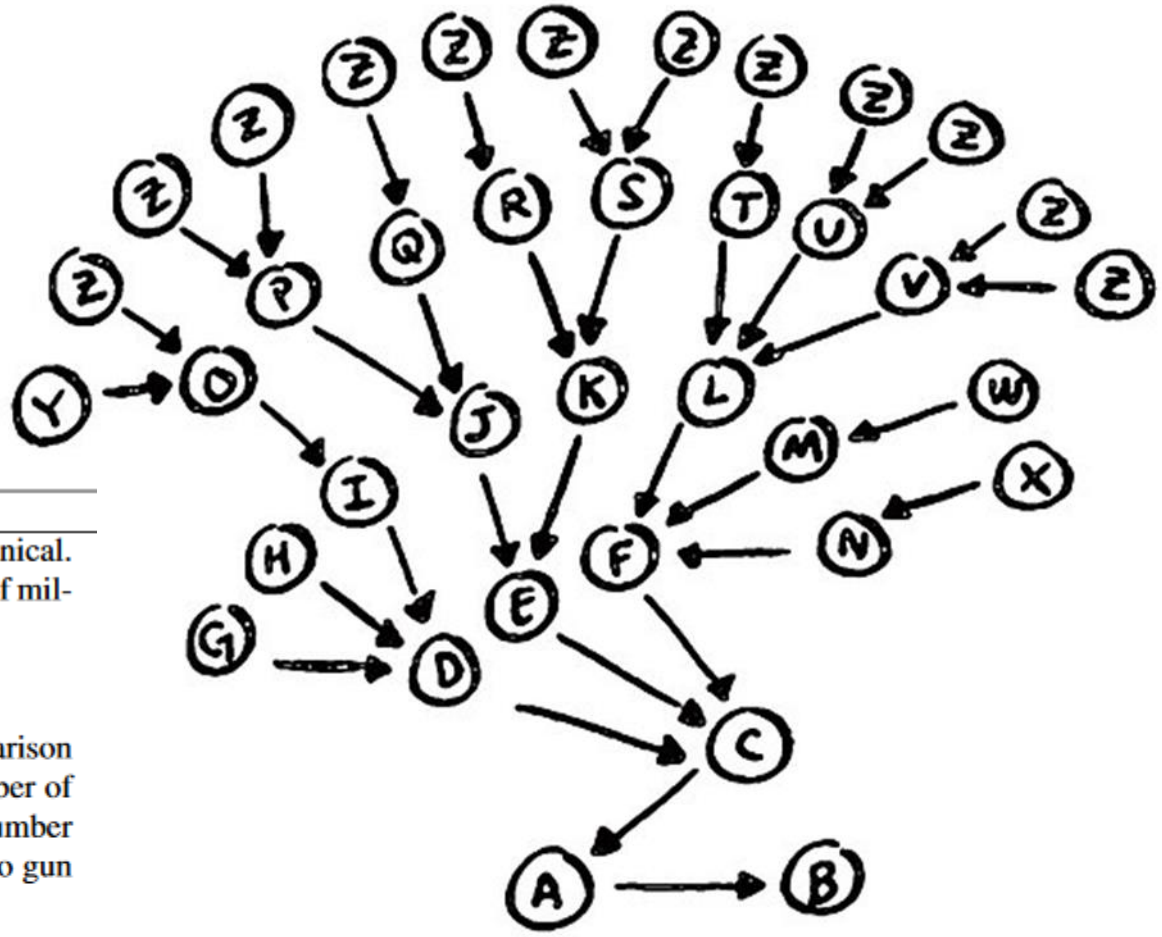
Yes, Russian Trolls Helped Elect Trump

Social media lies have real-world consequences.

DOTAN EXPOSED: MUSICIAN ACCUSED OF MASTERMINDING FAKE FAN ACCOUNTS

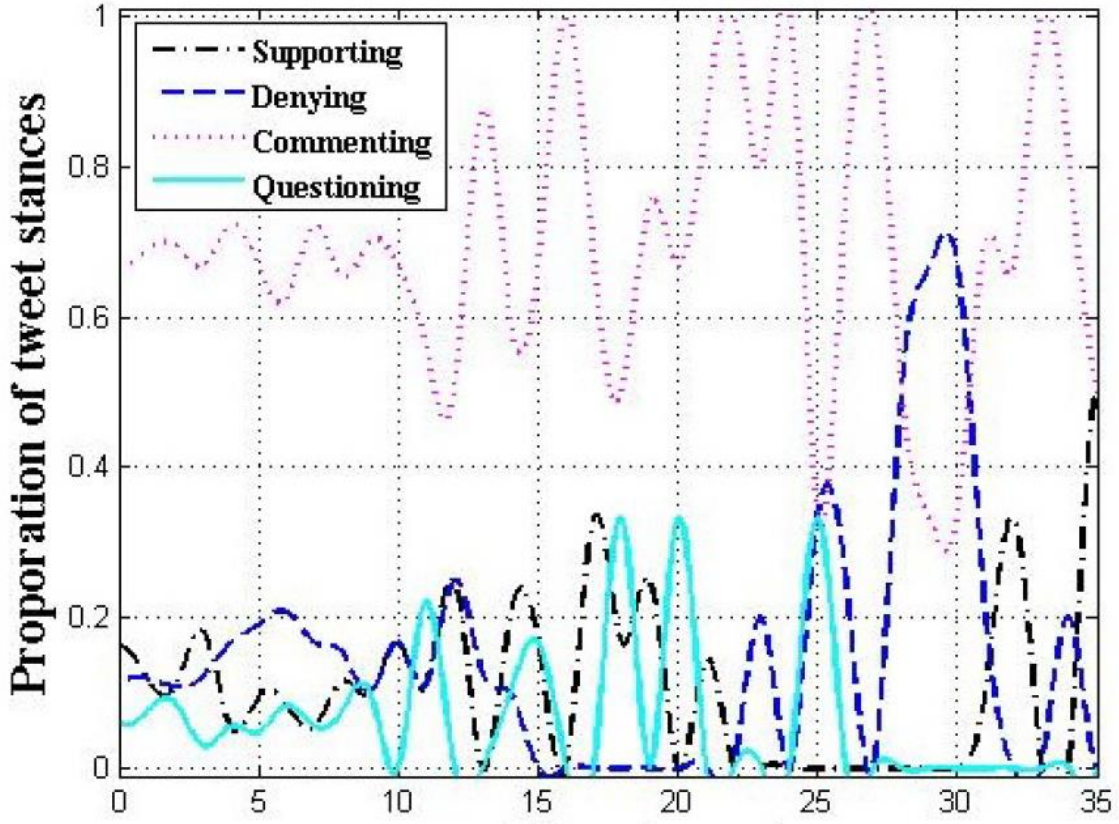
Stance detection

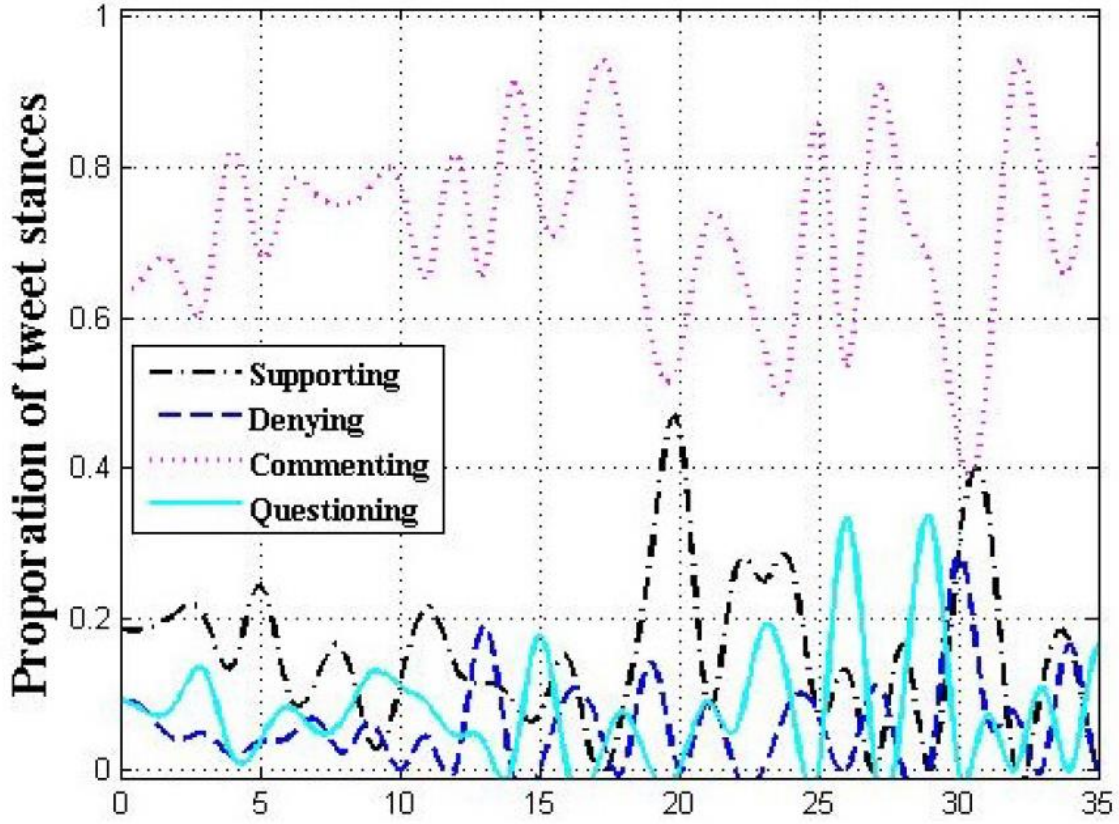
Class	Very High Degree	Neutral
Insult or Attack	Well, you have proven yoruself to be a man with no brain, that is for sure. The definition that was given was the one that scientists use, not the layperson.	The empire you defend is tyrannical. They are responsible for the death of millions.
	Is that what you said right before they started banning assault weapons?...Obviously, you're gullible. Since you're such a brainiac and all, why don't you visit the UN website and see what your beloved UN is up to?	Bad comparisons. A fair comparison would be comparing the total number of defensive gun uses to the total number of gun crimes (not just limiting it to gun homicides).
Sarcasm	My pursuit of happiness is denied by trees existing. Let's burn them down and destroy the environment. It's much better than me being unhappy.	An interesting analysis of that article you keep quoting from the World Net Daily [url]
	Like the crazy idea the Earth goes around the Sun.	Indeed there is no diffrence it is still a dead baby but throwing a baby in a trash can and leaving it for dead is far more cruel than abortion.



[1]

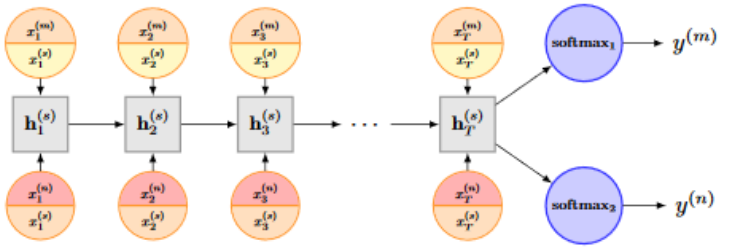
Truthfulness related to stances



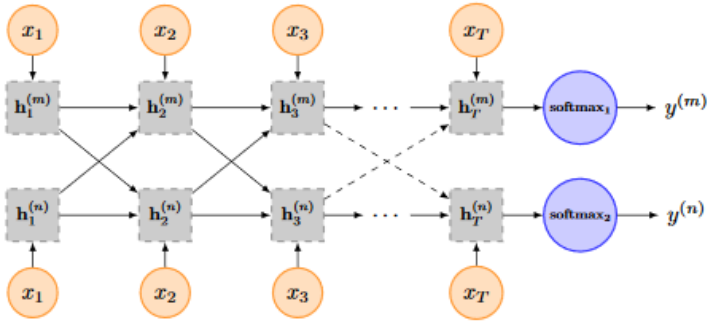


Related work

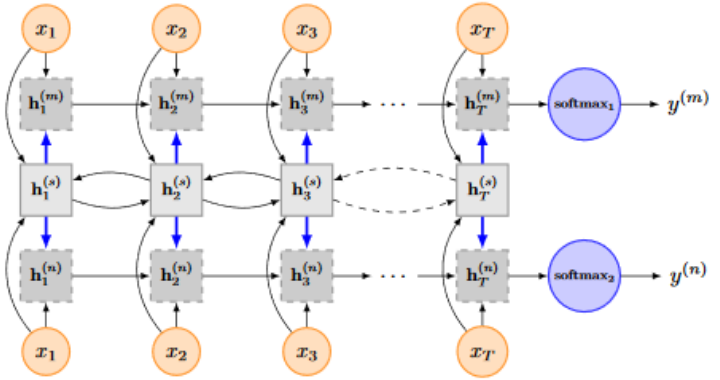
Multi-task learning on movie reviews [2]



(a) Model-I: Uniform-Layer Architecture



(b) Model-II: Coupled-Layer Architecture



(c) Model-III: Shared-Layer Architecture

Key Contributions

- Joint learning for rumor and stance detection
- Improvements on [2],
  - Heterogeneous rumor related tasks
  - Separate objectives of different tasks
  - Usage of GRU

Data

- Twitter dataset from Lui et al for Rumor detection
- News articles from Fake News Challenge for stance classification
- Twitter dataset from Pheme dataset for stance classification

Rumor detection

Claims $\{ C_1, C_2, ..., C_{|C|} \}$
Where each claim $C_i = \{(x_{ij}, t_{ij})\}$
Many-to-one:
$f : x_{i1}, x_{i2}, \ldots, x_{iT_i} \rightarrow Y_i$

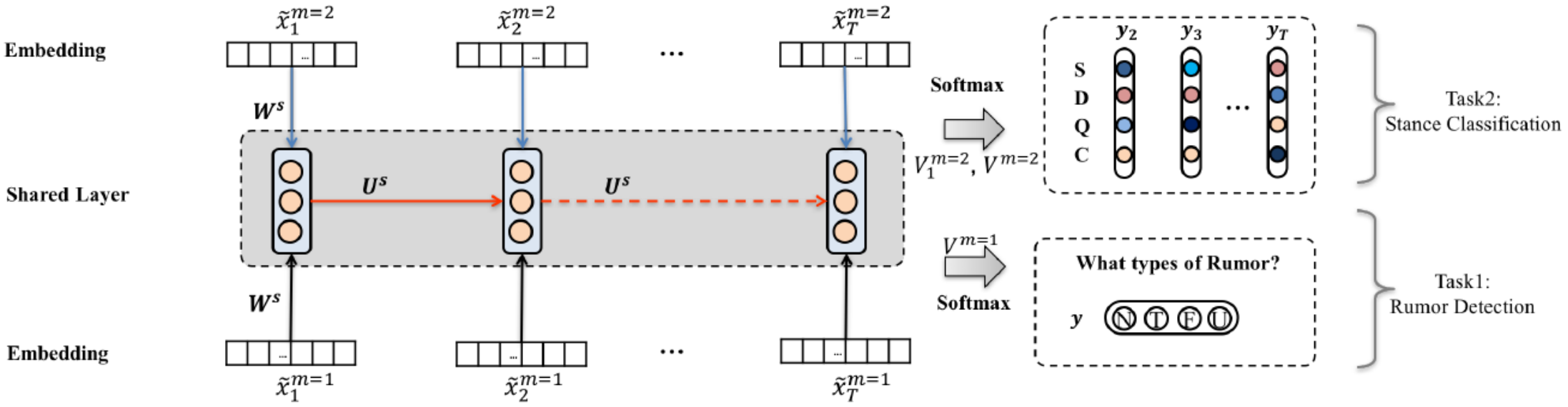
Y = {Non-rumor, True rumor, False rumor, Unverified rumor}

Stance detection

Claims $\{ C_1, C_2, ..., C_{|C|} \}$
Where each claim $C_i = \{(x_{ij}, t_{ij})\}$
Many-to-many:
$g : x_{i1}, x_{i2}, \ldots, x_{iT_i} \rightarrow Y_{i1}, Y_{i2}, ..., Y_{iT_i}$

Y = {Supporting, Denying, Questioning, Commenting}
or
Y = {Agree, Disagree, Discuss, Unrelated}

Model 1: Uniform Shared-Layer Architecture



(a) Uniform Shared-Layer Architecture

Standard GRU

$$z_t = \sigma\left(x_t U^z + h_{t-1} W^z\right)$$
$$r_t = \sigma\left(x_t U^r + h_{t-1} W^r\right)$$
$$\tilde{h}_t = \tanh\left(x_t U^h + (r_t * h_{t-1}) W^h\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Shared layer

$$\tilde{x}_t^m = E^m x_t^m$$

$$r_t^m = \sigma\left(W_r^s \tilde{x}_t^m + U_r^s h_{t-1}^m\right)$$

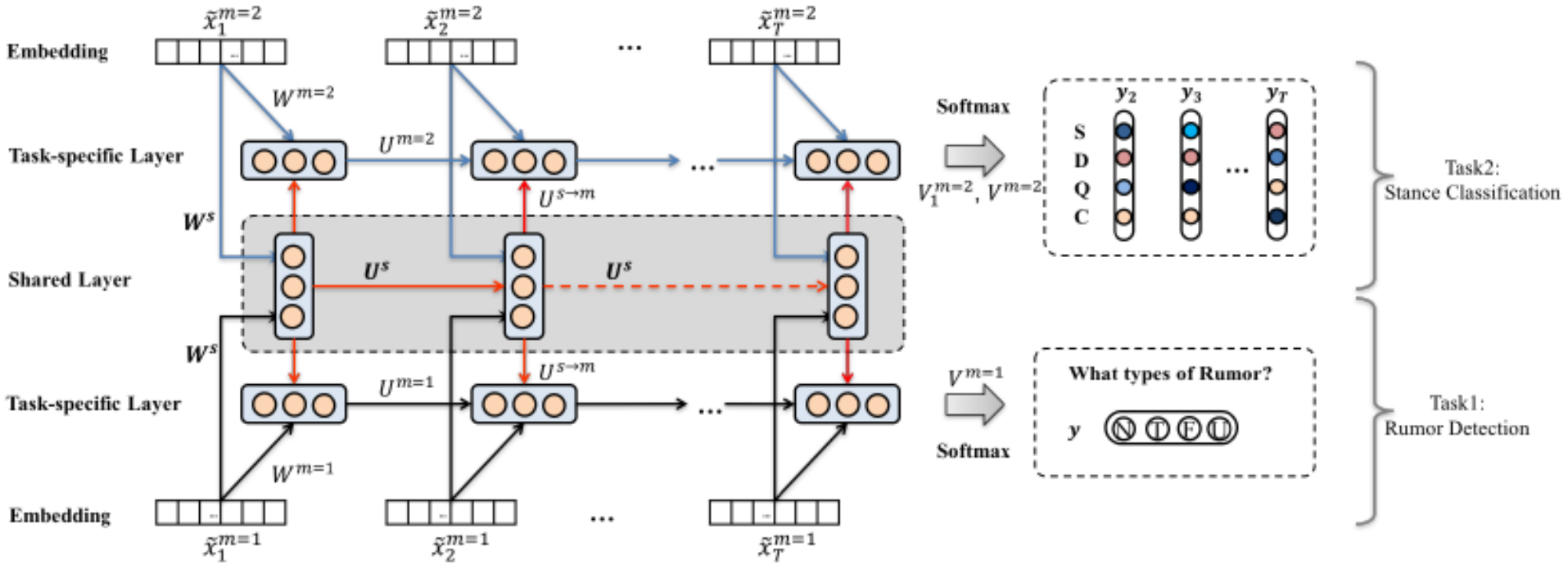
$$z_t^m = \sigma\left(W_z^s \tilde{x}_t^m + U_z^s h_{t-1}^m\right)$$

$$\tilde{h}_t^m = tanh\left(W_h^s \tilde{x}_t^m + U_h^s (h_{t-1}^m \odot r_t)\right)$$

$$h_t^m = (1 - z_t^m) \odot h_{t-1}^m + z_t^m \odot \tilde{h}_t^m$$

X uses vector representation with tf*idf values

Model 2: Enhanced Shared-Layer Architecture



(b) Enhanced Shared-Layer Architecture

Task specific layer

$$\tilde{x}_t^m = E^m x_t^m$$

$$r_t^m = \sigma\left(W_r^m \tilde{x}_t^m + U_r^m h_{t-1}^m + U_r^{s\to m} h_t^s\right)$$

$$z_t^m = \sigma\left(W_z^m \tilde{x}_t^m + U_z^m h_{t-1}^m + U_z^{s\to m} h_t^s\right)$$

$$\tilde{h}_t^m = tanh\left(W_h^m \tilde{x}_t^m + U_h^m (h_{t-1}^m \odot r_t^m) + U_h^{s\to m} h_t^s\right)$$

$$h_t^m = (1 - z_t^m) \odot h_{t-1}^m + z_t^m \odot \tilde{h}_t^m$$

Training

Multi task model

$$L = -\sum_t \sum_c g_t^c \log y_t^c + \lambda ||\Theta||_2^2$$

Input : A set of claims $\{C_1, C_2, \cdots, C_{|C|}\}$, ϵ

Initialize model parameters $\Theta = \{W^s, U^s, E^m, W^m, U^m, U^{s \to m}, V_1^m, V^m, b^m\}$ randomly;
for *iteration from 0 to maxIter* **do**
    1. Pick a task m randomly;
    2. Pick random training sample(s) from task m;
    3. Compute loss $L(\Theta)$ using Eq 5;
    4. Compute gradient $\nabla(\Theta)$;
    5. Update model: $\Theta \leftarrow \Theta - \epsilon \nabla(\Theta)$;
end

Experiments

LIU: Data distribution conform real world

(a) Rumor detection dataset

LIU+	N	T	F	U
Claim #	2,280	99	498	123
Proportion	76.0%	3.3%	16.6%	4.1%
posts # / Claim	757	1,029	587	686
Users #	61,7374	6,5475	18,2459	5,5298

(b) Stance classification dataset

PHEME	Support	Deny	Question	Comment
Tweets #	891	335	353	2,855
Proportion	20.09%	7.56%	7.96%	64.39%
Users #	732	295	318	2,036
FNC	Agree	Disagree	Discuss	Unrelated
articles #	5,581	1,537	13,373	54,894
Proportion	7.40%	2.03%	17.74%	72.81%
Sentence #	62,593	18,090	146,872	582,206

Results Rumor detection

Results depending on dataset other task
Positive influence of task specific layer

Method	MicF1	MacF1	N F_1	F F_1	T F_1	U F_1
DTR [47]	0.734	0.338	0.856	0.349	0.071	0.076
SVM-RBF [45]	0.760	0.216	0.864	0.000	0.000	0.000
DTC [6]	0.793	0.357	0.883	0.528	0.018	0.000
SVM-TS [33]	0.786	0.361	0.879	0.506	0.037	0.014
RFC [23]	**0.799**	0.389	**0.889**	0.541	0.031	0.091
MT-single [32]	0.762	0.426	0.875	0.487	0.05	0.292
	LIU+ & PHEME datasets					
MT-US	0.761	0.431	0.872	0.513	0.089	0.292
MT-ES	0.783	**0.464**	0.876	**0.534**	**0.114**	**0.333**
	LIU+ & FNC dataset					
MT-US	0.752	0.439	0.858	**0.545**	0.105	0.323
MT-ES	0.778	**0.443**	0.872	0.503	**0.074**	**0.324**

Results Stance

Results dependending on dataset
Disagree stands out

(a) PHEME dataset (S: Support; D: Deny; Q: Question; C: Comment)

Method	MicF1	MacF1	S F_1	D F_1	Q F_1	C F_1
Majority Vote	0.641	0.195	0.000	0.000	0.000	0.781
NB [38]	0.277	0.244	0.395	0.038	0.182	0.362
DT [18]	0.552	0.374	0.421	0.112	0.278	0.688
BOW [36]	**0.652**	0.344	0.273	0.108	0.206	**0.790**
HP[30]	0.650	0.390	0.519	0.079	0.394	0.771
CNN [7]	0.642	0.324	0.301	0.08	0.178	0.739
BiGRU [3]	0.605	0.373	0.299	0.158	0.286	0.751
MT-single	0.583	0.344	0.212	0.154	0.272	0.737
MT-US	0.635	0.400	**0.355**	0.116	0.337	0.776
MT-ES	0.622	**0.430**	0.314	**0.158**	**0.531**	0.739

(b) FNC dataset (A: Agree; N: Disagree; D: Discuss; U: Unrelated)

Method	MicF1	MacF1	A F_1	N F_1	D F_1	U F_1
Majority Vote	0.722	0.209	0.000	0.000	0.000	0.839
NB [38]	0.676	0.214	0.000	0.003	0.043	0.810
DT [18]	0.615	0.240	0.054	0.013	0.127	0.767
BOW [36]	**0.724**	0.214	0.010	0.000	0.000	**0.847**
HP[30]	–	–	–	–	–	–
CNN [7]	0.691	0.277	0.054	0.000	0.242	0.817
BiGRU [3]	0.571	0.305	0.178	0.025	0.297	0.718
MT-single	0.584	0.291	0.163	0.026	0.243	0.731
MT-US	0.604	0.310	0.094	**0.103**	**0.298**	0.741
MT-ES	0.609	**0.328**	**0.219**	0.096	0.251	0.744

Layer behavior

Model	Shared Layer	Rumor-specific	Stance-specific
MT-ES	really?, what? not like, great, omg disgusting, scary I guess, probably	what?, really? is real/fact totally false seriously wrong	why?, what is what happened no doubt, may not sure, really?
MT-single	–	what is, what? seriously wrong totally false is real, wtf?	no doubt may be, not what happened what is, why

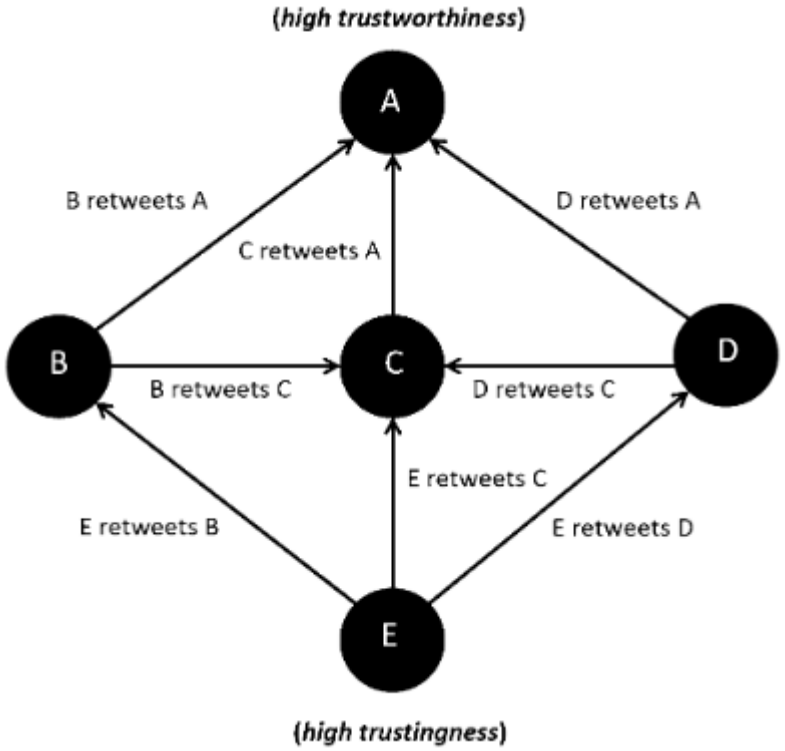
Conclusion

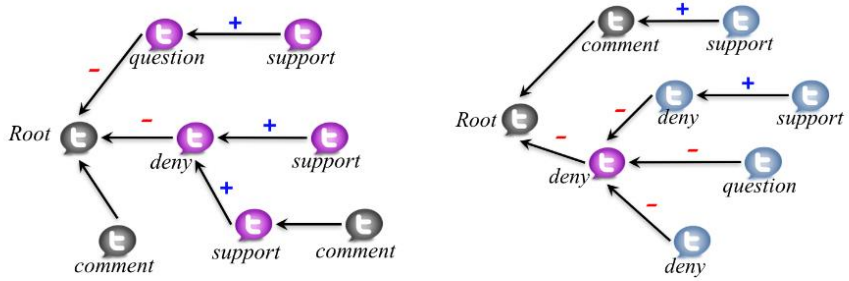
Positive effect of joint learning with multi-task model
Dataset of the other task influences current task

Future work Rumor Detection

Utilizing computational trust to identify rumor spreaders [3]

Rumor Detection on Twitter with Tree-structured Recursive NeuralNetworks [4]





(a) False rumor (b) True rumor

Future work Stance Classification

From Stances' Imbalance to Their Hierarchical Representation and Detection [5]

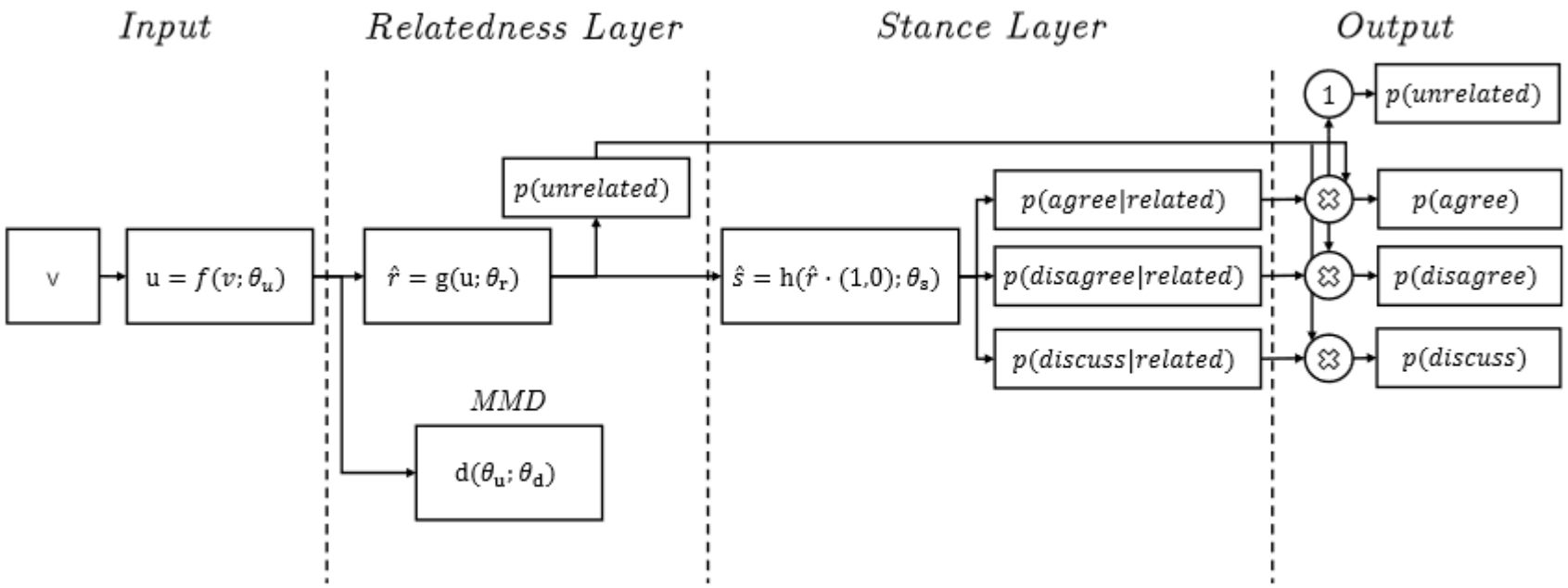


Figure 1: The architecture of our proposed two-layer neural network.

My thoughts

Real world problem
Implementation seemed very simple
Multi task learning with N tasks
Usage of Twitter data

References

1. Abbott, R., Walker, M., Anand, P., Fox Tree, J. E., Bowmani, R., & King, J. (2011, June). How can you say such things?!?: Recognizing disagreement in informal political argument. In *Proceedings of the Workshop on Languages in Social Media* (pp. 2-11). Association for Computational Linguistics.
2. Liu, P., Qiu, X., & Huang, X. (2016). Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.
3. Rath, B., Gao, W., Ma, J., & Srivastava, J. (2018). Utilizing computational trust to identify rumor spreaders on Twitter. *Social Network Analysis and Mining*, *8*(1), 64.
4. Ma, J., Gao, W., & Wong, K. F. (2018, July). Rumor detection on twitter with tree-structured recursive neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1980-1989).
5. Zhang, Q., Liang, S., Lipani, A., Ren, Z., & Yilmaz, E. (2019, May). From Stances' Imbalance to Their Hierarchical Representation and Detection. In *Companion Proceedings of the The Web Conference*.