LogDTL: Network Log Template Generation with Deep Transfer Learning

Thieu Nguyen[1], Satoru Kobayashi[2], and Kensuke Fukuda[2,*]

nguyenthieu2102@gmail.com, sat@nii.ac.jp , kensuke@nii.ac.jp

[1]Hanoi University of Science and Technology, Hanoi, Vietnam

[2]National Institute of Informatics, Tokyo, Japan

System logs (syslog)

- System Logs is generated by the operating system components, softwares, programs about device changes, device drivers, system changes, events, operations…
- System Logs (Syslog) are used in many important network systems such as: Network devices, routers, etc.

```
May 17 11:15:00 192.168.100.3 [Error] bgp_read_packet error: Connection reset by peer
```

||

Error about BGP packet from 192.168.100.3 occurred.
BGP connection to 192.168.100.3 was reset by peer.

Log format

- Syslog: Free-format description (due to developer's convenience and flexibility)
  - →Hard to extract
    - + manually information from logs due to the large amount of logs
    - + automatically information due to the data format
  - →Challenge: generated tons of logs (eg: SINET4 <u>70k</u> logs/day average)

- Log Template consists of **descriptions** and **variables**

Notification sent to neighbor 192.168.100.2: type 6/3

↓

Notification sent to neighbor ** : type **/**

Description

Variable

Description

Variable

The advantages of Log Template

- Decrease the number of logs to check
  - Can classify logs with log templates

Notification sent to neighbor 192.168.100.2: type 6/3
Notification sent to neighbor 192.168.100.3: type 6/3
stream_read_try: read failed on fd 9: Connection reset by peer
192.168.100.3 [Error] bgp_read_packet error: Connection reset by peer

Notification sent to neighbor 192.168.100.2: type 6/3
Notification sent to neighbor 192.168.100.3: type 6/3

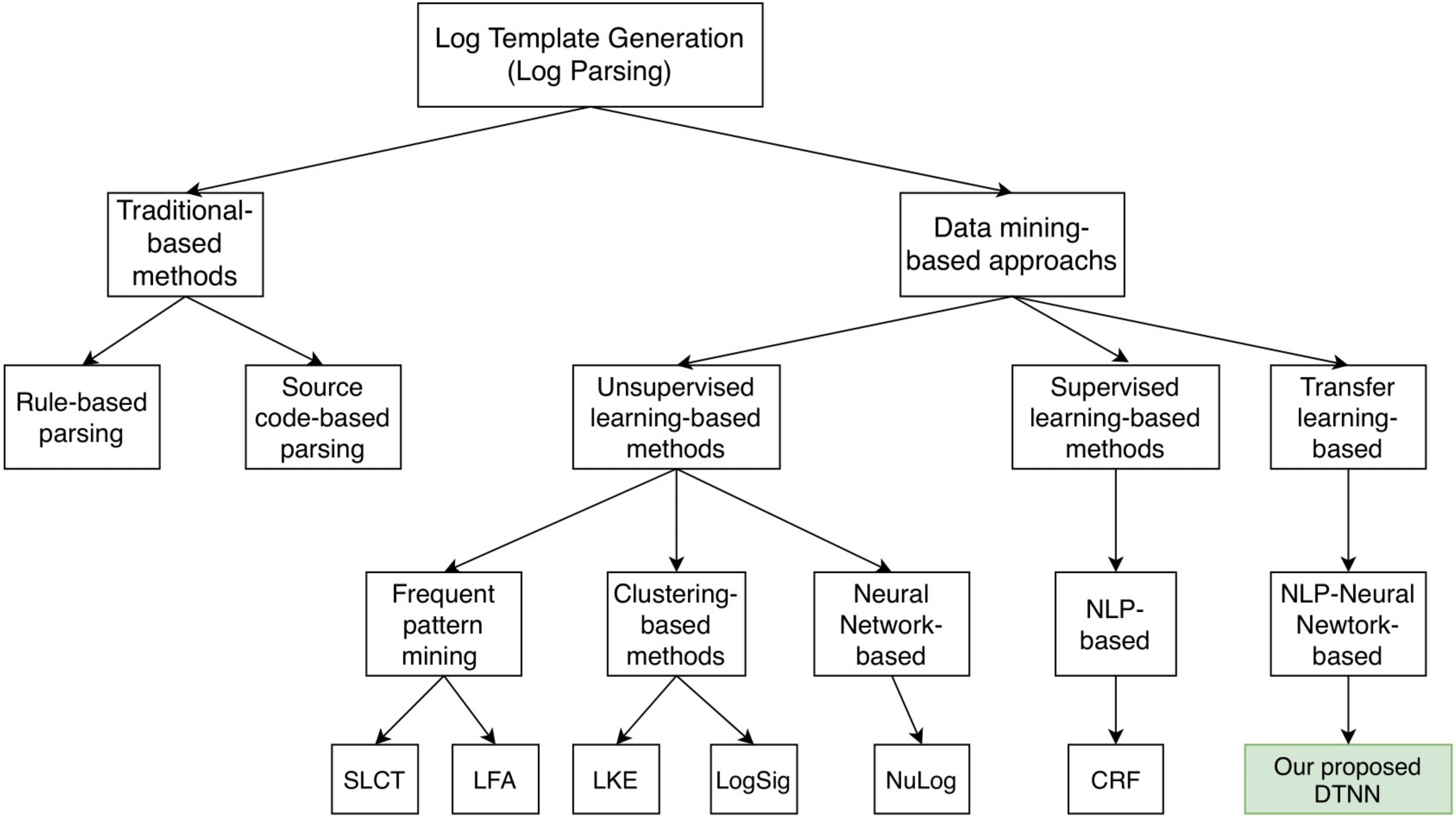
stream_read_try: read failed on fd 9: Connection reset by peer

192.168.100.3 [Error] bgp_read_packet error: Connection reset by peer

Error!

- Find time-series events among log templates
  - Causal relationship estimation
  - Anomaly detection
- The question is: How can we make the log template for the system?

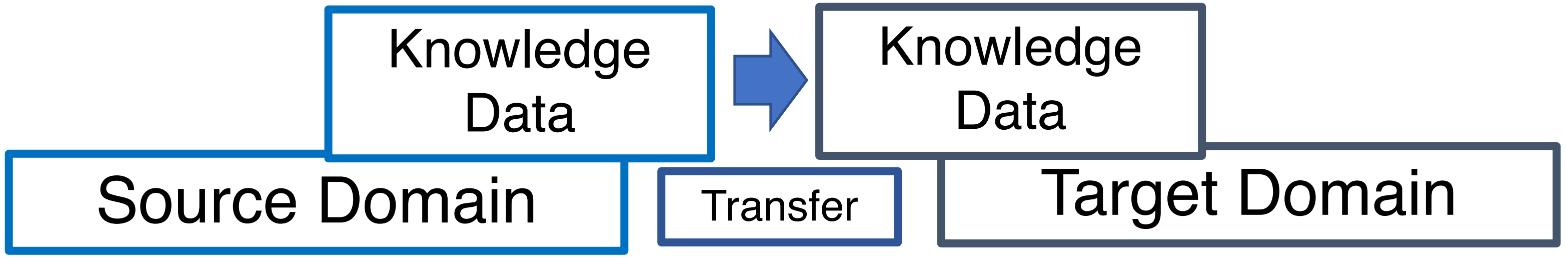
Overview of past literature



Pros and Cons of past literature

- Traditional way:
  - time-consuming and error-prone pain
  - manually writing ad-hoc rules to parse a huge volume of logs
- Data mining-based:
  - NOT need any software information
  - Unsupervised learning-based:
    - Difficult to distinguish Variable from Description automatically, therefor low accuracy
  - <u>Supervised learning-based</u>:
    - High accuracy but require more labeling training data
- How?
  - Using fewer logs and less human power
  - Generate correct log templates
- Idea: Transfer Learning

What is Transfer Learning?



- Similar to target topic
- But not the same
- Sufficient knowledge

- Target topic
- Want to solve this
- Insufficient knowledge

Transferring knowledge/data of source domain,
solve the problem of target domain with a high precision[3]

Transfer Learning for log template generation

vyatta

bgpd %ADJCHANGE: neighbor ** Down **

- Source domain
  - Open source software
    - can make correct log templates from source code
    - can make learning data easily from log templates
- Target domain
  - Proprietary software

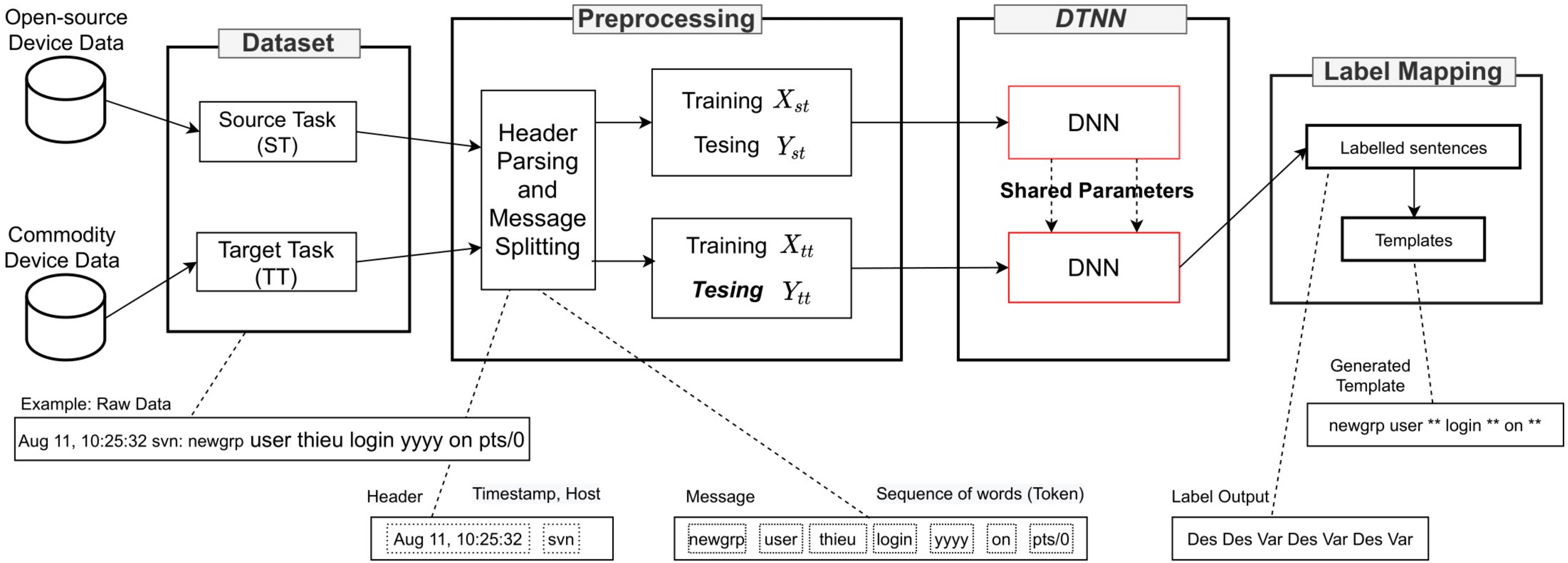
junos

%BGP-5-ADJCHANGE neighbor ** ** BGP Notification sent

- Noted: Source and target usually follow common network protocols

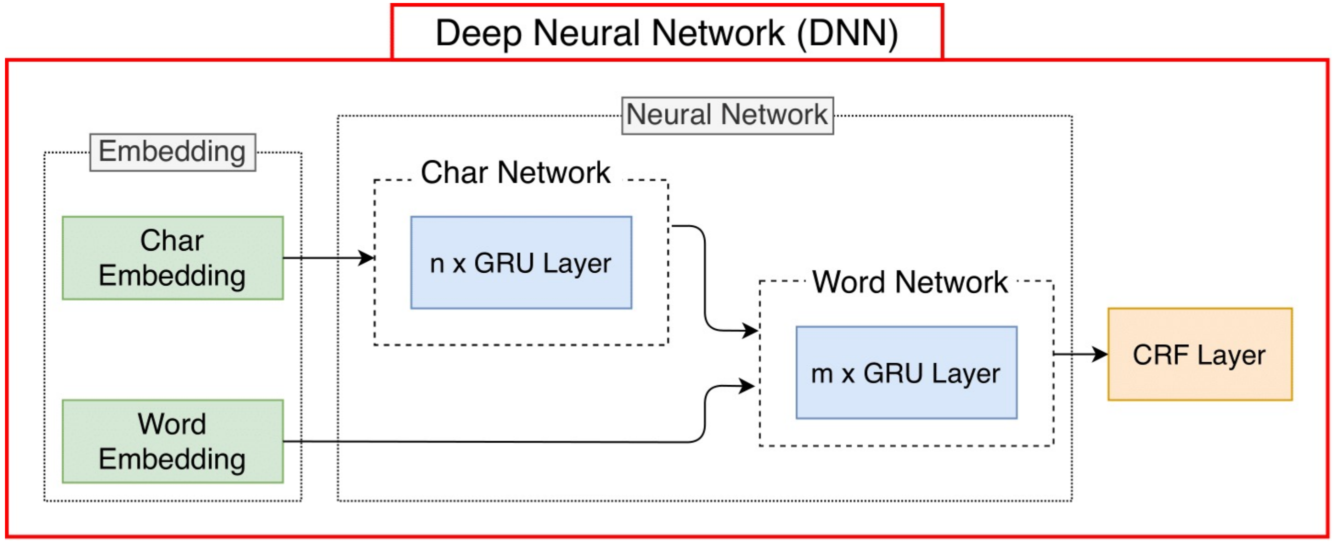
- **Transfer learning solves the problems**[4]:
  → Make learning data for proprietary software.
  → Using fewer log-data and handicraft work
  → Solve the target task problems with high precision

Architecture overview



LogDTL: Log template generation using Deep Transfer Learning

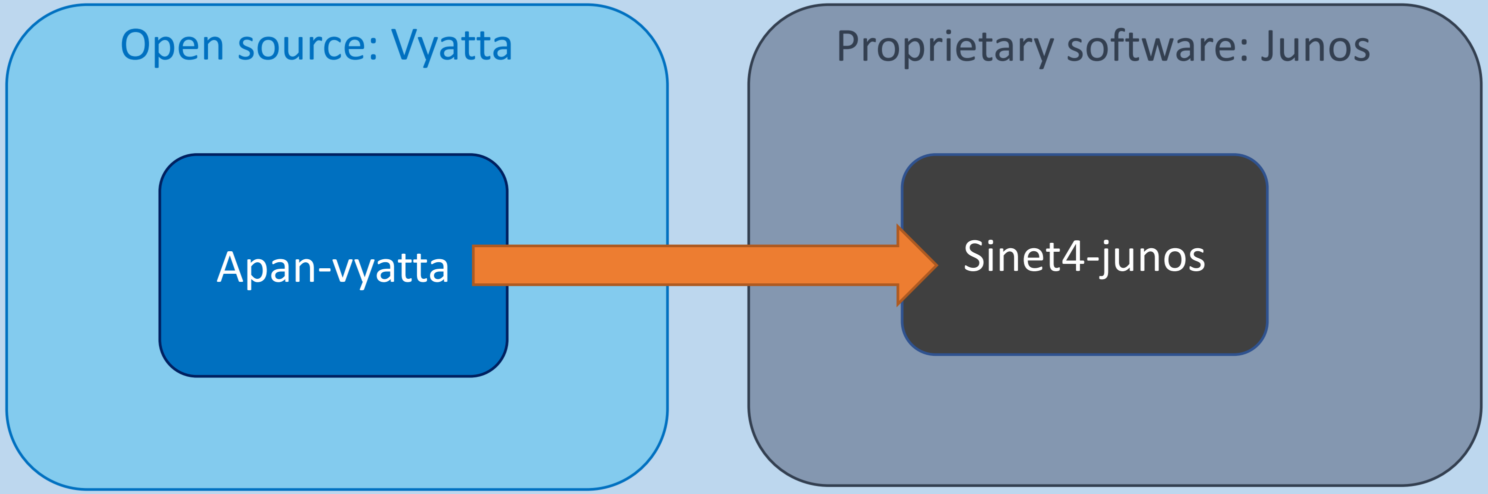
Deep transfer neural network



- Based on three ideas:
  - Extension of simple Conditional Random Field (CRF)[5]
  - Transfer Learning based on Deep Neural Network[6]
  - Semantic with word-level and character-level in NLP
- Transfer learning happens by transfer the knowledge (the weights of network) learned from source domain and apply it to target domain

[5] Satoru Kobayashi (2014), [6] Sebastian Ruder (2019)

Dataset overview (source and target)



Transfer Apan-vyatta -> sinet4-junos

- Different OS (vyatta vs junos)
- Same function (routing)
- Same network protocol (e.g., BGP, OSPF, ...)
- Same data type (syslog)

Training and testing dataset

Attributes	Source Task (AV dataset) (Open source software)		Target Task (S4 dataset) (Proprietary software)	
	Training (SX)	Testing (SY)	Training (TX)	Testing (TY)
#Sentences	100000	100000	74496	74910
#Clusters	38	45	41	75

- The 1st goal is to generate log templates for proprietary network equipment based on transfer learning → log template of S4 dataset will be learned from known templates and features of AV dataset.
- We sampled 1, 10, 100, 1000, 10000 training data (TX) of target task
  - To validate how many labeled training data do we need to prepare

Baselines and metrics

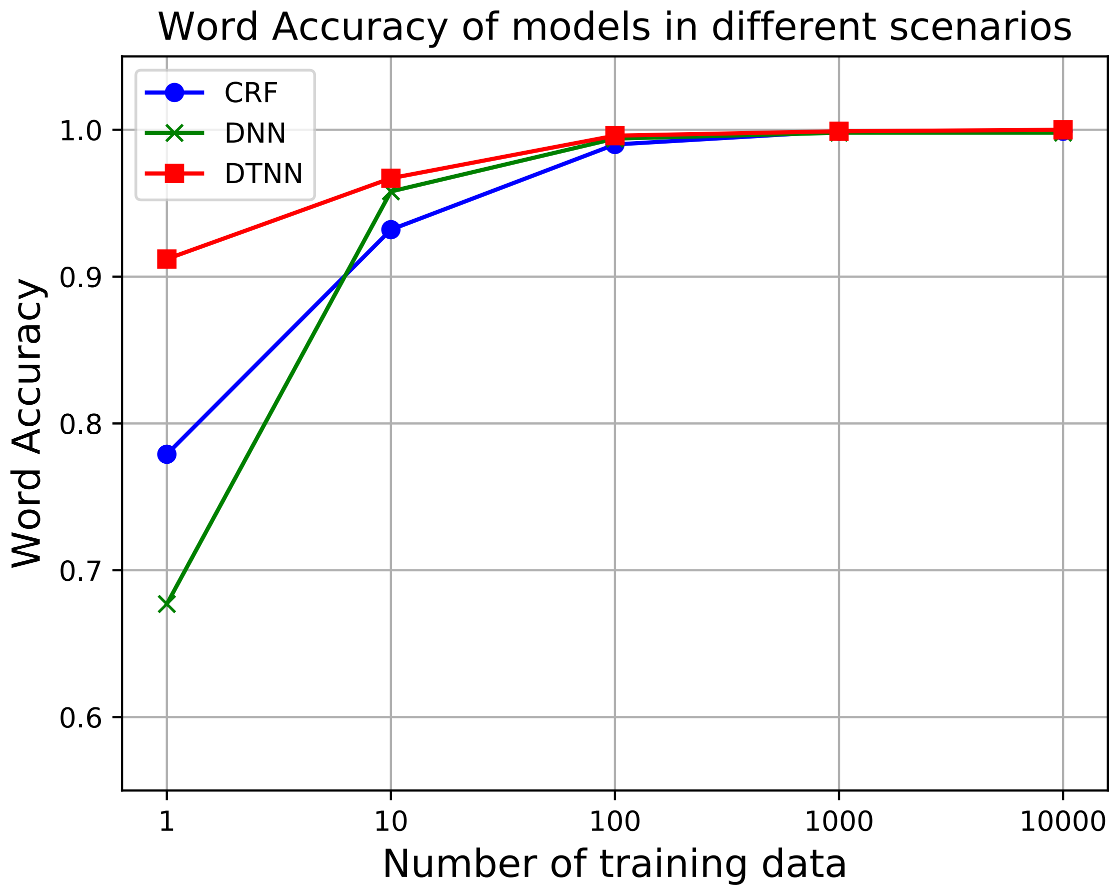
$$\mathrm{W}A = \frac{\sum_{i=1}^{N} W_i^C}{\sum_{i=1}^{N} W_i}$$

$$\mathrm{TW}A = \frac{\sum_{i=1}^{N} W_i^C}{N}$$

- **Comparison models**
  - Simple Conditional Random Field (CRF)
  - Deep Neural Network (DNN)
  - Deep Transfer Neural Network (DTNN)

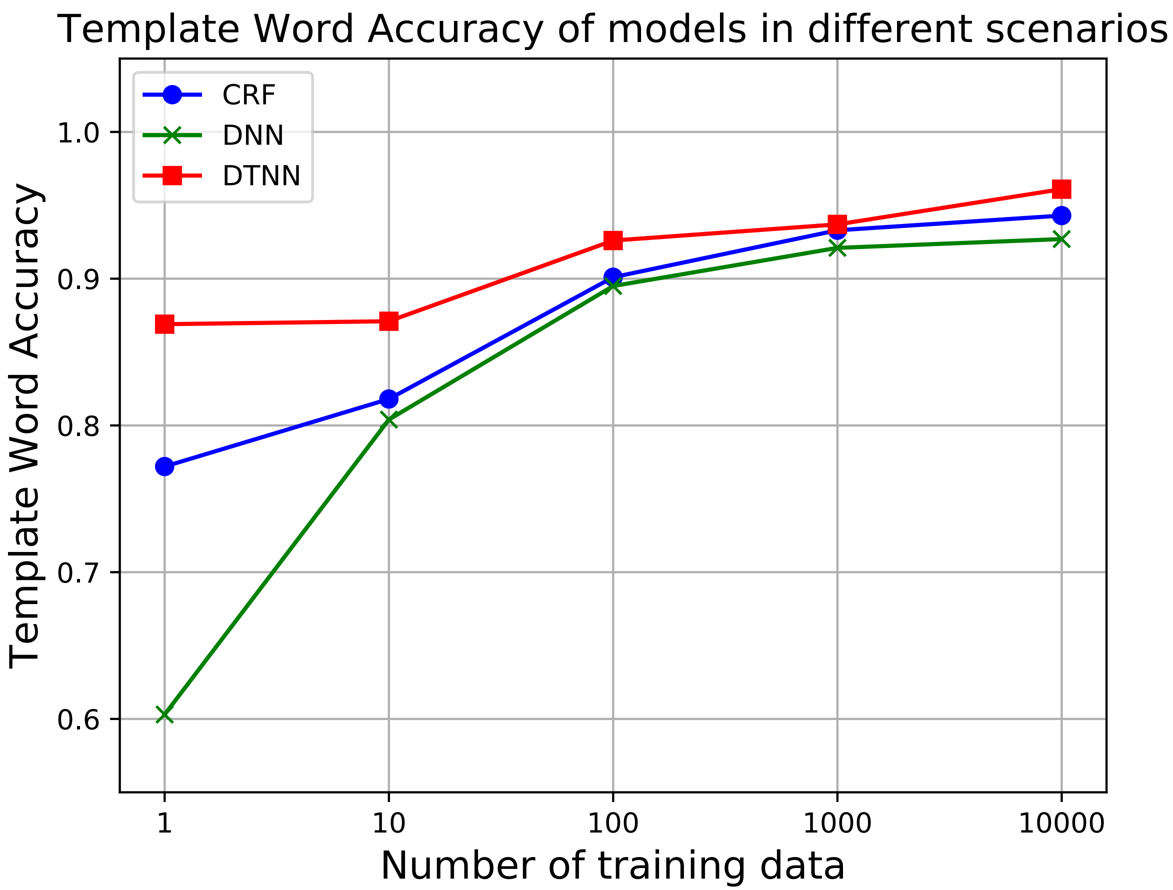
- **Performance metrics**
  - Word Accuracy (WA): simply count the number of correctly predicted word (W_i^C) per the total of word in testing dataset.
  - Template Word Accuracy (TWA): measure the average of the scores for each log templates.
  - Both WA and TWA evaluate whether each word is labeled (i.e., classified as Description or Variable) correctly or not.

Word Accuracy



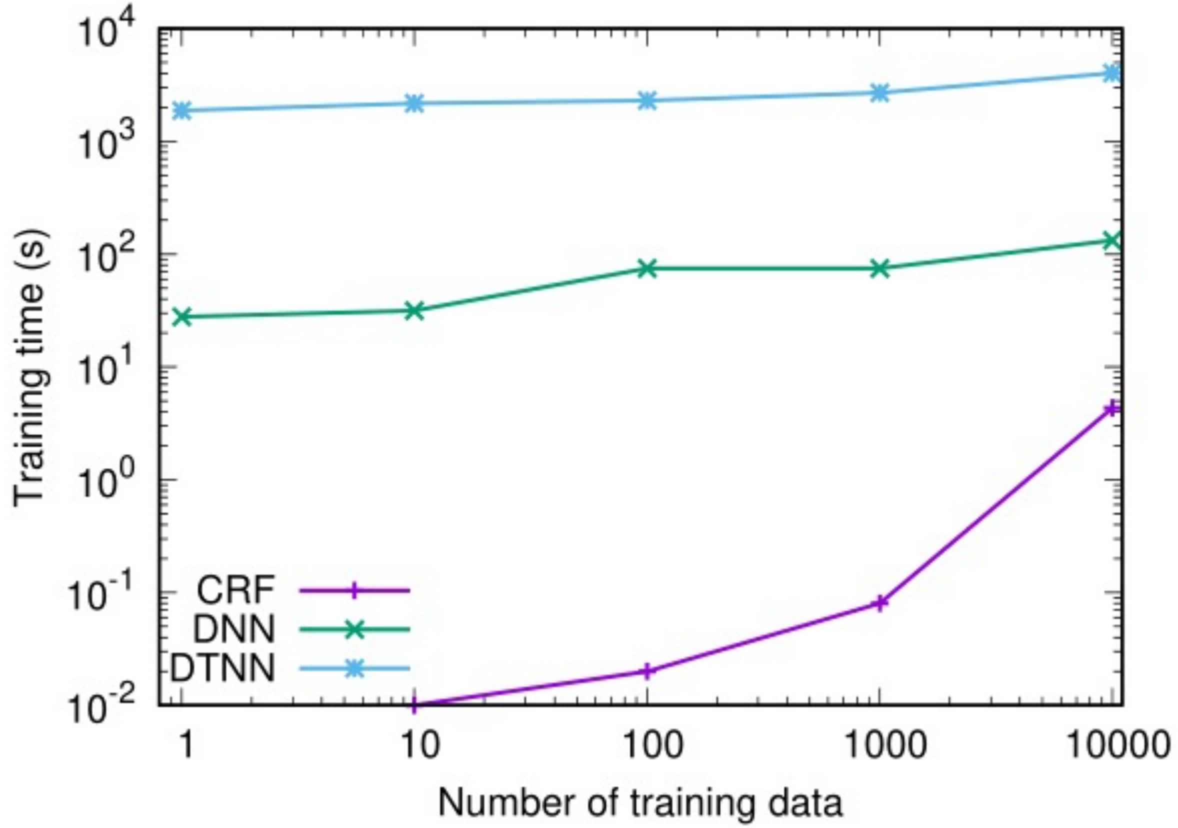
- DTNN is better than both DNN and CRF in most cases, even with low number of training data.
- CRF performs better than DNN with lower training data, but when having enough training data, both CRF and DNN get similar results.

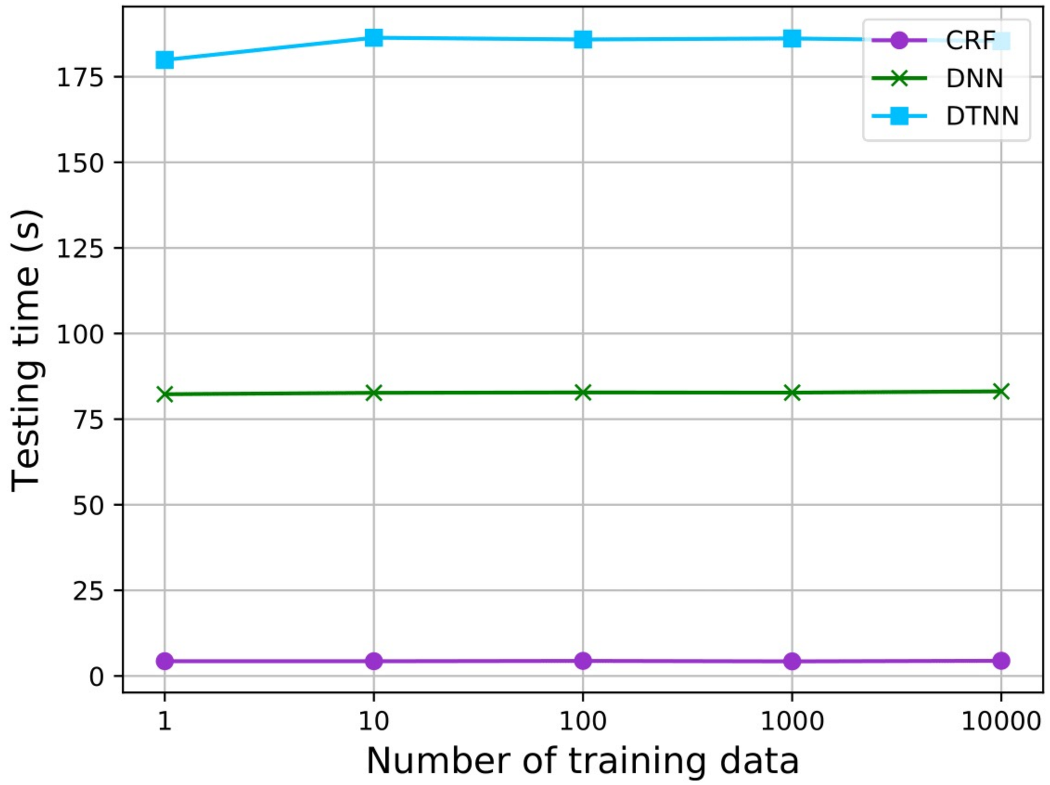
Template Word Accuracy



- DTNN outperform both DNN and CRF in all cases.
- CRF is better DNN in all cases.

Training and Testing time





Case Study

Method	Template 1	Template 2
Ground truth	rpd ** EVENT MTU ** index ** Up Broadcast P2P Multicast addr ** **	/kernel MTU for ** reduced to **
CRF	rpd ** EVENT MTU (ifname) index ** Up Broadcast P2P Multicast addr ** (v6 addr)	/kernel MTU for (v6 addr) reduced to **
DNN	** ** EVENT MTU ** ** ** Up Broadcast ** ** ** (num) **	/kernel MTU for ** reduced to **
DTNN	rpd ** EVENT MTU ** index ** Up Broadcast P2P Multicast addr ** **	/kernel MTU for ** reduced to **

- Our proposed DTNN be able to predict correctly both template 1 and 2
- DNN predicted template 2 correctly but wrongly with template 1
- CRF failed to predict both templates.

Conclusion

- **Proposal: Proprietary software's log-template estimation with deep transfer learning**
  - An extension of simple Conditional Random Field (CRF)
  - Transfer Learning based on Deep Neural Network
  - Semantic with both word-level and character-level in NLP
- **Good result even with different domains**
  - Our proposed DTNN outperforms cutting-edge DNN and CRF models with different test cases and different performance metrics.
  - DTNN works in reasonable processing time for testing phase
  - Could use DTNN model to generated labeling training dataset for other models.
- **Future works**
  - Investigating the validity of our approach for the larger class of log template categories
- **https://github.com/fukuda-lab/LogDTL**

Thank you for listening!

This work is supported by MIC/SCOPE #191603009.