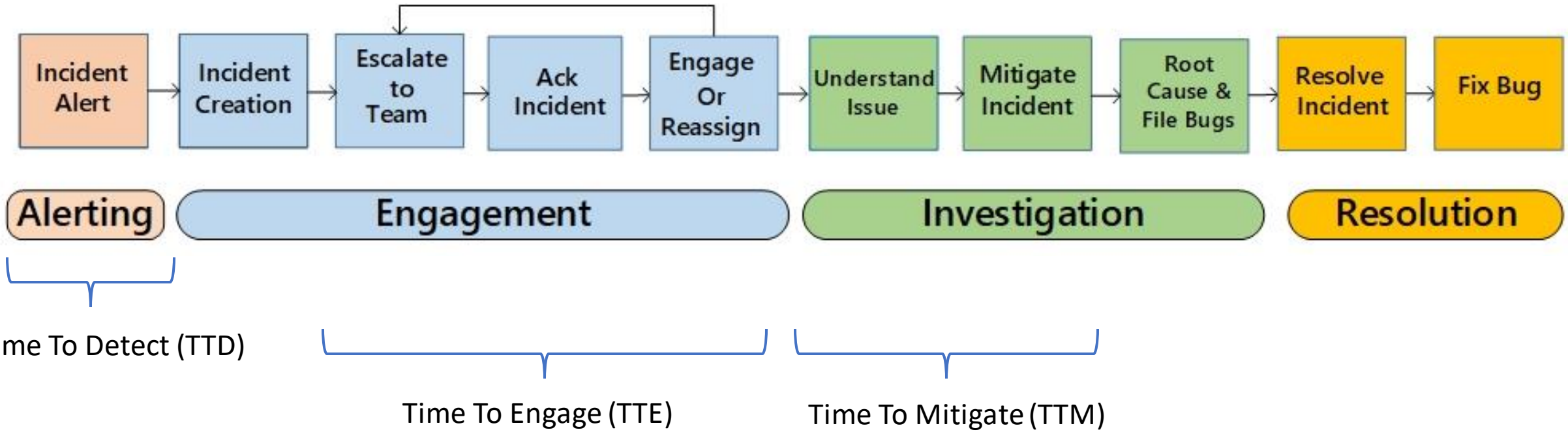
Neural Knowledge Extraction from Cloud Service Incidents

Manish Shetty, Chetan Bansal, Sumit Kumar, Nikitha Rao,
Nachiappan Nagappan, Thomas Zimmermann

Microsoft

Incident Life-Cycle



Motivation

		Data Center Management	Network	Storage	Compute	Database	Web Services
Severity	Critical	38.33x	8.46x	10.06x	142.05x	209.97x	286.6x
	High	19.25x	3.18x	2.52x	2.56x	5.75x	3.56x
	Medium	1x	9.8x	7.09x	2.95x	25.28x	12.93x
	Low	3.01x	5.49x	1.09x	11.65x	2.41x	144.79x

Distribution of relative incident fixing time for Microsoft core services*

** Incident Fixing Time = TTF = TTD +TTE + TTM*

* Chen, Zhuangbin, et al. "Towards intelligent incident management: why we need it and how we make it." Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. 2020.

Problem

Incident Manager | Dashboard | Resources

Title: Firewall: Need help determining which configuration rule allows access from source IP 127.0.0.1

Status: **Resolved**
Severity: Medium
Summary:

Approver: alice@company.com

General Details

Subscription ID: c398a7aa-9b69-4d1b-8838-9fb20490e2a1
ASC link with details: https://service.company.com/myexplorer/linktoresource/~2A~-532e-4e96-8e78-aaa104b16cd3~d2resource~

Description of Problem:
The issue is that customer saw traffic allowed from an IP: 127.0.0.1 even though there is no rule allowing the traffic. We were able to verify that the traffic is allowed. Essentially, the question is this: Is it possible to check what was the rule that triggered to allow traffic inbound from source ip, when no rule was configured to allow that traffic?

Linux ASM Syslog not showing any relevant info

ADDITIONAL INFORMATION FROM SUPPORT CENTER

Lack of any
Schema / Structure

*Soft*NER

- We frame the knowledge extraction problem as a **Named-Entity Recognition** task.
- We have built the SoftNER framework, for unsupervised knowledge extraction.

Entity Name	Example
Problem Type	VNet Failure
Exception Message	The vpn gateway deployment operation failed due to an intermittent error
Failed Operation Name	Create and Mount Volume
Resource Id	/resource/2aa3abc0-7986-1abc-a98b-443fd7245e6f/resourcegroups/cs-net/providers/network/frontdoor/
Tenant Id	4536dcd6-e2e1-3465-a22b-d25f62456233
Vnet Id	45ea1234-123b-7969-adaf-e0255045569e
Link With Details	https://supportcenter.cloudx.com/caseoverview?srid=112
Device Name	sab01-98cba-1d
Source IP	198.168.0.1
Status Code	500
Location	eastus2

Example of entities extracted by SoftNER

Related Work – Incident Management

FSE 2020

Identifying Linked Incidents in Large-Scale Online Service Systems

Yujun Chen*
Microsoft Research
Beijing, China

Xian Yang
Hong Kong Baptist University
Hong Kong, China

Hang Dong
Microsoft Research
Beijing, China

Xiaoting He*
Chinese Academy of Sciences
Beijing, China

Hongyu Zhang
The University of Newcastle
NSW, Australia

Qingwei Lin†
Microsoft Research
Beijing, China

Junjie Chen
College of Intelligence and Computing, Tianjin University
Tianjin, China

Pu Zhao
Yu Kang†
Microsoft Research
Beijing, China

Feng Gao
Zhangwei Xu
Microsoft Azure
Redmond, USA

Dongmei Zhang
Microsoft Research
Beijing, China

ICSE 2019

An Empirical Investigation of Incident Triage for Online Service Systems

Junjie Chen[1,2], Xiaoting He[3], Qingwei Lin[3], Yong Xu[3], Hongyu Zhang[4], Dan Hao[1,2]
Feng Gao[5], Zhangwei Xu[5], Yingnong Dang[5], Dongmei Zhang[3]
[1]Key Laboratory of High Confidence Software Technologies (Peking University), MoE
[2]Institute of Software, EECS, Peking University, Beijing, 100871, China
{chenjunjie,haodan}@pku.edu.cn
[3]Microsoft Research, Beijing 100080, China,{v-xiah,qlin,yox,dongmeiz}@microsoft.com
[4]The University of Newcastle, NSW 2308, Australia, hongyu.zhang@newcastle.edu.au
[5]Microsoft, Redmond, WA 98052, USA,{fgao,zhangxu,yidang}@microsoft.com

***Abstract*—Online service systems have become increasingly popular. During operation of an online service system, incidents (unplanned interruptions or outages of the service) are inevitable. As an initial step of incident management, it is important to be able to automatically assign an incident report to a suitable team. We call this step *incident triage*, which can significantly affect the efficiency and accuracy of overall incident management. To better understand the incident-triage practice in industry, we perform an empirical study of incident triage on 20 large-scale online**

average cost of service downtime has steadily increased from $505,502 in 2010 to $740,357 in 2016[2].

Once an incident of an online service system occurs, it needs to be mitigated as soon as possible. The goal is to minimize the service downtime and to ensure high quality of the provided service. Currently, incident management has become a critical task for online service systems. A typical procedure of incident management is as follows. When an incident is detected by

How to Mitigate the Incident? An Effective Troubleshooting Guide Recommendation Technique for Online Service Systems

Jiajun Jiang*
College of Intelligence and Computing, Tianjin University
Tianjin 300350, China
jiangjiajun@tju.edu.cn

Weihai Lu
School of Software and Microelectronics, Peking University
Beijing 100871, China
luweihai@pku.edu.cn

Junjie Chen
College of Intelligence and Computing, Tianjin University
Tianjin 300350, China
junjiechen@tju.edu.cn

Qingwei Lin
Microsoft Research
Beijing 100080, China
qlin@microsoft.com

Pu Zhao
Microsoft Research
Beijing 100080, China
pu.zhao@microsoft.com

Yu Kang
Microsoft Research
Beijing 100080, China
kay@microsoft.com

Hongyu Zhang
The University of Newcastle
NSW 2308, Australia
hongyu.zhang@newcastle.edu.au

Yingfei Xiong
Key Laboratory of High Confidence Software Technologies(MoE), DCST
PKU, Beijing 100871, China
xiongyf@pku.edu.cn

Feng Gao
Microsoft Azure
Redmond, WA 98052, USA
fgao@microsoft.com

FSE 2020

Towards Intelligent Incident Management: Why We Need It and How We Make It

Zhuangbin Chen*
The Chinese University of Hong Kong
Hong Kong, China

Yu Kang†
Liqun Li
Xu Zhang
Microsoft Research, China

Hongyu Zhang
The University of Newcastle
NSW, Australia

Hui Xu
Yangfan Zhou
Fudan University
Shanghai, China

Li Yang
Jeffrey Sun
Microsoft Azure, USA

Zhangwei Xu
Yingnong Dang
Feng Gao
Microsoft Azure, USA

Pu Zhao
Bo Qiao
Qingwei Lin†
Dongmei Zhang
Microsoft Research, China

Michael R. Lyu
The Chinese University of Hong Kong
Hong Kong, China

ABSTRACT

The management of cloud service incidents (unplanned interruptions or outages of a service/product) greatly affects customer satisfaction and business revenue. After years of efforts, cloud enterprises are able to solve most incidents automatically and timely. However, in practice, we still observe critical service incidents

ACM Reference Format:
Zhuangbin Chen, Yu Kang, Liqun Li, Xu Zhang, Hongyu Zhang, Hui Xu, Yangfan Zhou, Li Yang, Jeffrey Sun, Zhangwei Xu, Yingnong Dang, Feng Gao, Pu Zhao, Bo Qiao, Qingwei Lin, Dongmei Zhang, and Michael R. Lyu. 2020. Towards Intelligent Incident Management: Why We Need It and How We Make It. In *Proceedings of the 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*

FSE 2020

Related Work – Incident Diagnosis

KDD 2014

Correlating Events with Time Series for Incident Diagnosis

Chen Luo*
Jilin University
rackingroll@163.com

Jian-Guang Lou
Microsoft Research
jlou@microsoft.com

Qingwei Lin
Microsoft Research
qlin@microsoft.com

Qiang Fu
Microsoft Research
qifu@microsoft.com

Rui Ding
Microsoft Research
juding@microsoft.com

Dongmei Zhang
Microsoft Research
dongmeiz@microsoft.com

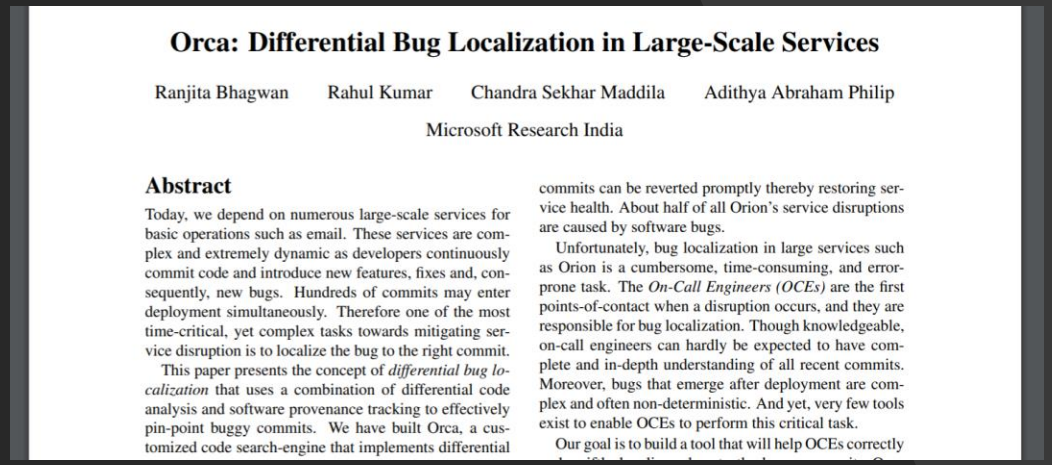
Zhe Wang
Jilin University
wz2000@jlu.edu.cn

ABSTRACT

As online services have more and more popular, incident diagnosis has emerged as a critical task in minimizing the service downtime and ensuring high quality of the services provided. For most online services, incident diagnosis is mainly conducted by analyzing a large amount of telemetry data collected from the services at runtime. Time series data and event sequence data are two major types of telemetry data. Techniques of correlation analysis are important tools that are widely used by engineers for data-driven in-

all time(24x7). However, during the operation of an online service, live-site service incidents (unplanned interruption/outage of the service) are often unavoidable, and can lead to significant economic loss or other serious consequences. For example, many reputable online services such as those provided by Amazon, Google, and Citrix have experienced live-site incidents during the past couple of years [1, 14]. In order to minimize service downtime caused by service incidents, much effort has been invested in improving the efficiency of service-incident diagnosis.

OSDI 2018



Orca: Differential Bug Localization in Large-Scale Services

Ranjita Bhagwan Rahul Kumar Chandra Sekhar Maddila Adithya Abraham Philip

Microsoft Research India

Abstract

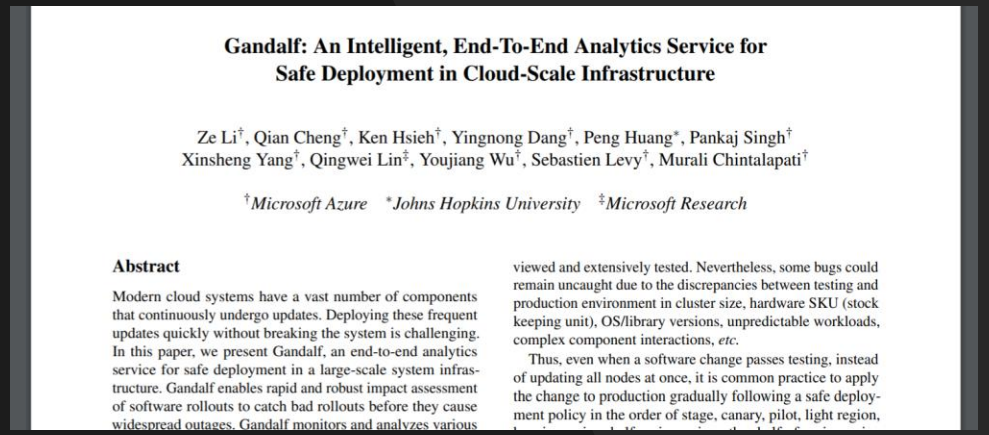
Today, we depend on numerous large-scale services for basic operations such as email. These services are complex and extremely dynamic as developers continuously commit code and introduce new features, fixes and, consequently, new bugs. Hundreds of commits may enter deployment simultaneously. Therefore one of the most time-critical, yet complex tasks towards mitigating service disruption is to localize the bug to the right commit.

This paper presents the concept of *differential bug localization* that uses a combination of differential code analysis and software provenance tracking to effectively pin-point buggy commits. We have built Orca, a customized code search-engine that implements differential

commits can be reverted promptly thereby restoring service health. About half of all Orion's service disruptions are caused by software bugs.

Unfortunately, bug localization in large services such as Orion is a cumbersome, time-consuming, and error-prone task. The *On-Call Engineers (OCEs)* are the first points-of-contact when a disruption occurs, and they are responsible for bug localization. Though knowledgeable, on-call engineers can hardly be expected to have complete and in-depth understanding of all recent commits. Moreover, bugs that emerge after deployment are complex and often non-deterministic. And yet, very few tools exist to enable OCEs to perform this critical task.

Our goal is to build a tool that will help OCEs correctly



Gandalf: An Intelligent, End-To-End Analytics Service for Safe Deployment in Cloud-Scale Infrastructure

Ze Li†, Qian Cheng†, Ken Hsieh†, Yingnong Dang†, Peng Huang*, Pankaj Singh†
Xinsheng Yang†, Qingwei Lin‡, Youjiang Wu†, Sebastien Levy†, Murali Chintalapati†

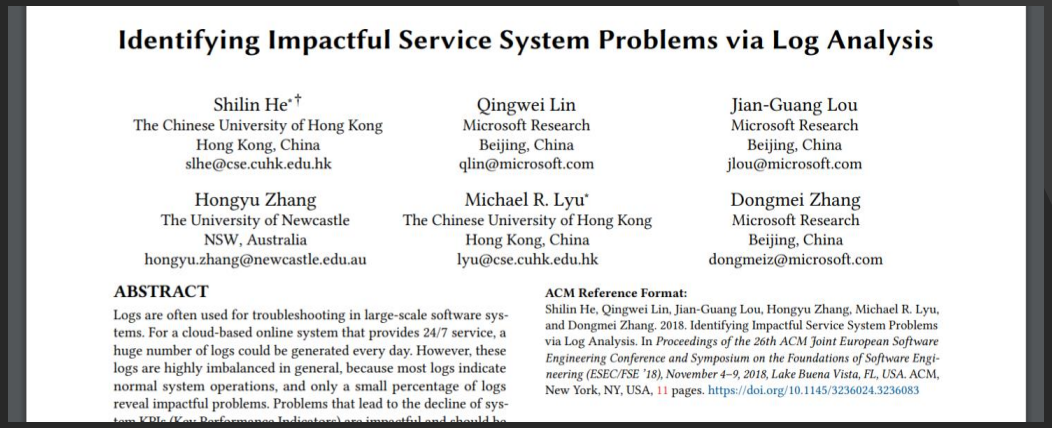
†*Microsoft Azure* **Johns Hopkins University* ‡*Microsoft Research*

Abstract

Modern cloud systems have a vast number of components that continuously undergo updates. Deploying these frequent updates quickly without breaking the system is challenging. In this paper, we present Gandalf, an end-to-end analytics service for safe deployment in a large-scale system infrastructure. Gandalf enables rapid and robust impact assessment of software rollouts to catch bad rollouts before they cause widespread outages. Gandalf monitors and analyzes various

viewed and extensively tested. Nevertheless, some bugs could remain uncaught due to the discrepancies between testing and production environment in cluster size, hardware SKU (stock keeping unit), OS/library versions, unpredictable workloads, complex component interactions, *etc.*

Thus, even when a software change passes testing, instead of updating all nodes at once, it is common practice to apply the change to production gradually following a safe deployment policy in the order of stage, canary, pilot, light region,



Identifying Impactful Service System Problems via Log Analysis

Shilin He*†
The Chinese University of Hong Kong
Hong Kong, China
slhe@cse.cuhk.edu.hk

Qingwei Lin
Microsoft Research
Beijing, China
qlin@microsoft.com

Jian-Guang Lou
Microsoft Research
Beijing, China
jlou@microsoft.com

Hongyu Zhang
The University of Newcastle
NSW, Australia
hongyu.zhang@newcastle.edu.au

Michael R. Lyu*
The Chinese University of Hong Kong
Hong Kong, China
lyu@cse.cuhk.edu.hk

Dongmei Zhang
Microsoft Research
Beijing, China
dongmeiz@microsoft.com

ABSTRACT

Logs are often used for troubleshooting in large-scale software systems. For a cloud-based online system that provides 24/7 service, a huge number of logs could be generated every day. However, these logs are highly imbalanced in general, because most logs indicate normal system operations, and only a small percentage of logs reveal impactful problems. Problems that lead to the decline of sys-

ACM Reference Format:
Shilin He, Qingwei Lin, Jian-Guang Lou, Hongyu Zhang, Michael R. Lyu, and Dongmei Zhang. 2018. Identifying Impactful Service System Problems via Log Analysis. In *Proceedings of the 26th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '18), November 4–9, 2018, Lake Buena Vista, FL, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3236024.3236083

NSDI 2018

FSE 2018

Related Work – Data Extraction

Snorkel Framework from Stanford

Snorkel: Rapid Training Data Creation with Weak Supervision

Alexander Ratner Stephen H. Bach Henry Ehrenberg
Jason Fries Sen Wu Christopher Ré
Stanford University
Stanford, CA, USA
{ajratner, bach, henryre, jfries, senwu, chrismre}@cs.stanford.edu

ABSTRACT

Labeling training data is increasingly the largest bottleneck in deploying machine learning systems. We present Snorkel, a first-of-its-kind system that enables users to train state-of-the-art models without hand labeling any training data. Instead, users write labeling functions that express arbitrary heuristics, which can have unknown accuracies and correlations. Snorkel denoises their outputs without access to ground truth by incorporating the first end-to-end implementation of our recently proposed machine learning paradigm, data programming. We present a flexible interface layer for writing labeling functions based on our experience over the past year collaborating with companies, agencies, and research labs. In a user study, subject mat-

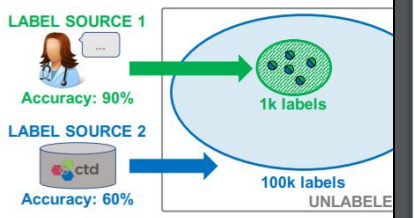


Figure 1: In Example 1.1, training data
by sources of differing accuracy and covera
key challenges arise in using this weak su
effectively. First, we need a way to estimat

VLDB 2018

SWELLSHARK: A Generative Model for Biomedical Named Entity Recognition without Labeled Data

Jason Fries, Sen Wu, Alex Ratner, Christopher Ré
Stanford University / Stanford, CA
{jfries,senwu,ajratner,chrismre}@cs.stanford.edu

Abstract

We present SWELLSHARK, a framework for building biomedical named entity recognition (NER) systems quickly and without hand-labeled data. Our approach views biomedical resources like lexicons as function primitives for autogenerating *weak supervision*. We then use a generative model to unify and denoise this supervision and

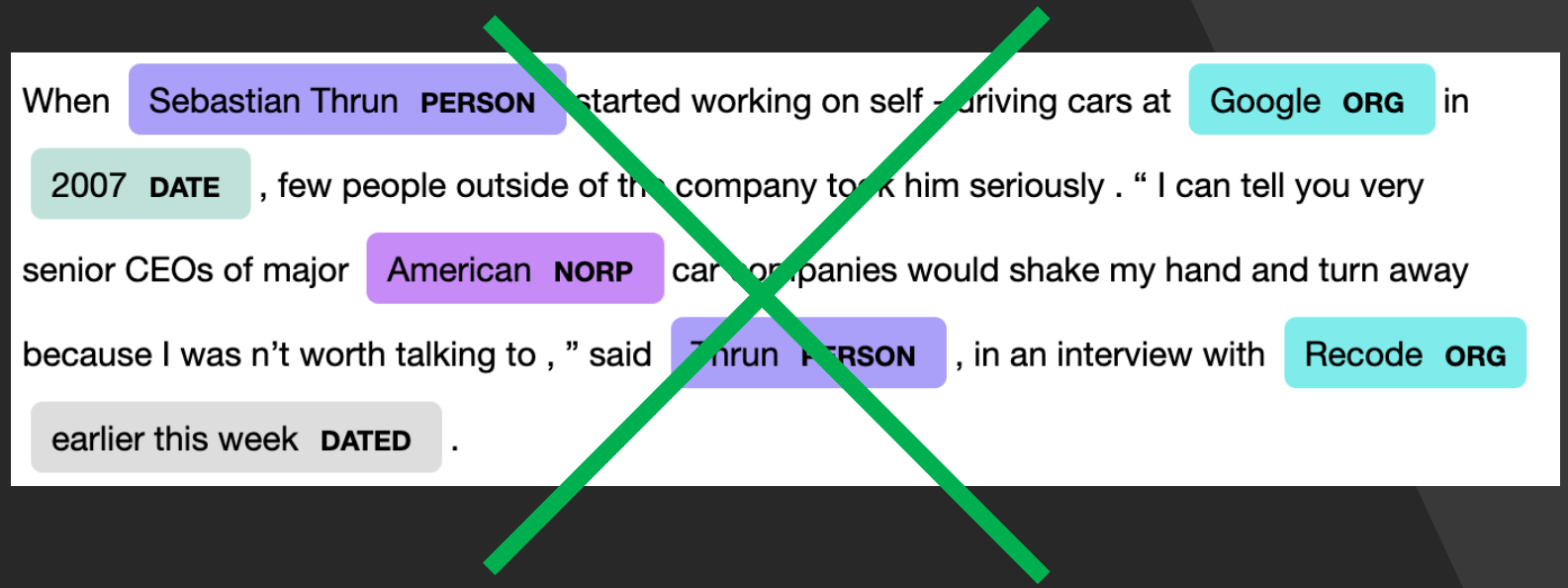
privacy concerns preventing distribution (Sabou et al., 2012; Gokhale et al., 2014). Furthermore, even expert inter-annotator agreement rates can be low for certain tasks.

In NLP, another common approach is *distant supervision* (Mintz et al., 2009) where structured resources like ontologies and knowledge bases are used to heuristically label training data. While noisy, this technique has shown empirical success. Distant supervision is commonly used with

Unique Aspects of DevOps Artifacts

1

100s of services and organization -> No fixed set of entities.



Unique Aspects of DevOps Artifacts

100s of services and organization -> No fixed set of entities.

1

```
"fault_type",
"node_id",
"correlation_guid",
"fault_code",
"memory",
"not_initialize_memory",
"hyper_v_error",
"instance_name",
```

HostOS

```
"vnet_id",
"mac_address",
"tenant_id",
"resource_uri",
"device_name",
"deployment_stage",
"default_interface_addresses",
"v_net_name",
"destination_ip",
"remote_port_range",
"tunnel_name",
"rack_id",
```

Azure Networking

Unique Aspects of DevOps Artifacts

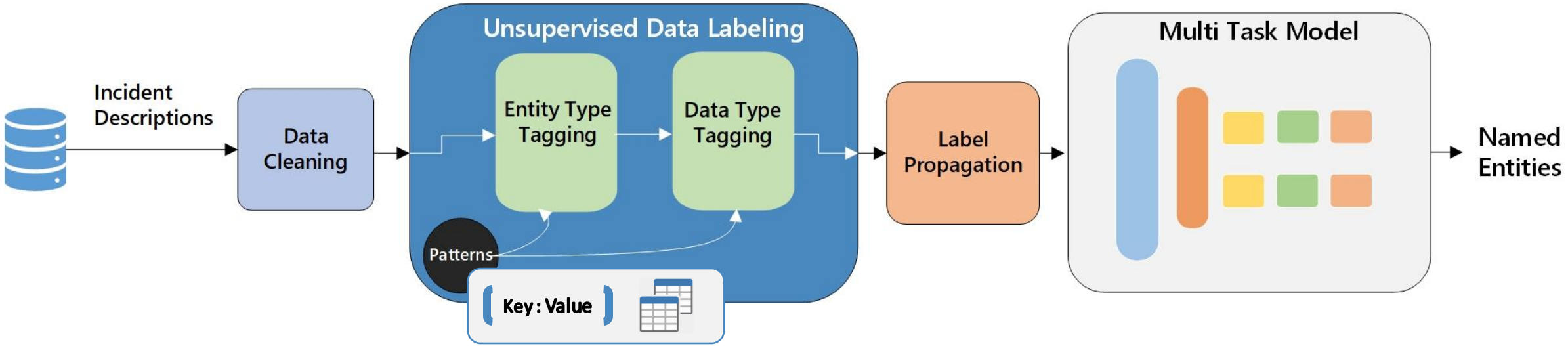
Large varying set of entities -> Expensive to obtain labelled data

Unique Aspects of DevOps Artifacts

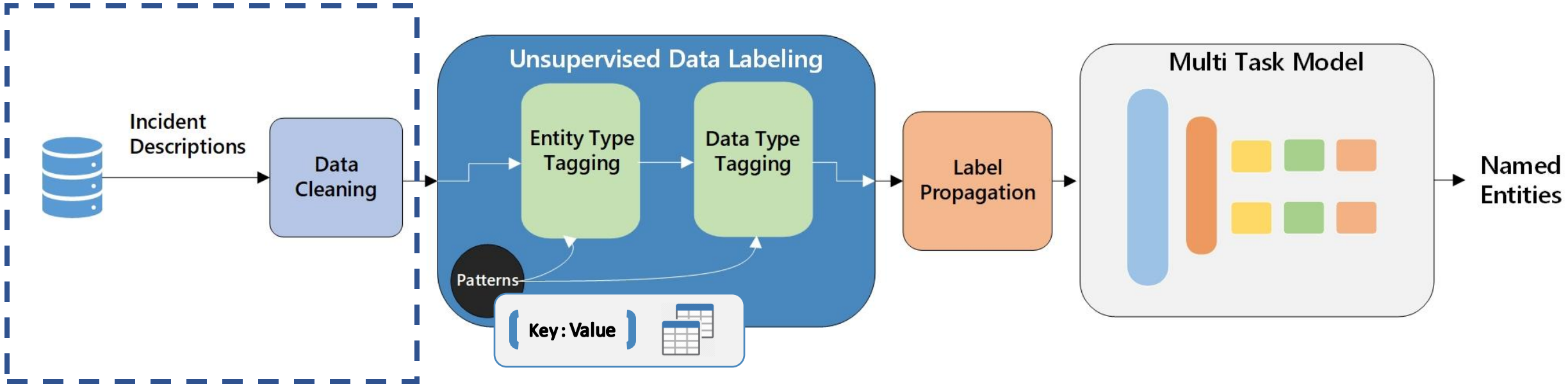
Unlike other domains, there is a rich set of types embedded with Natural Language.

Error Code
Stack Trace
URI
GUID
Ip Address
Exception

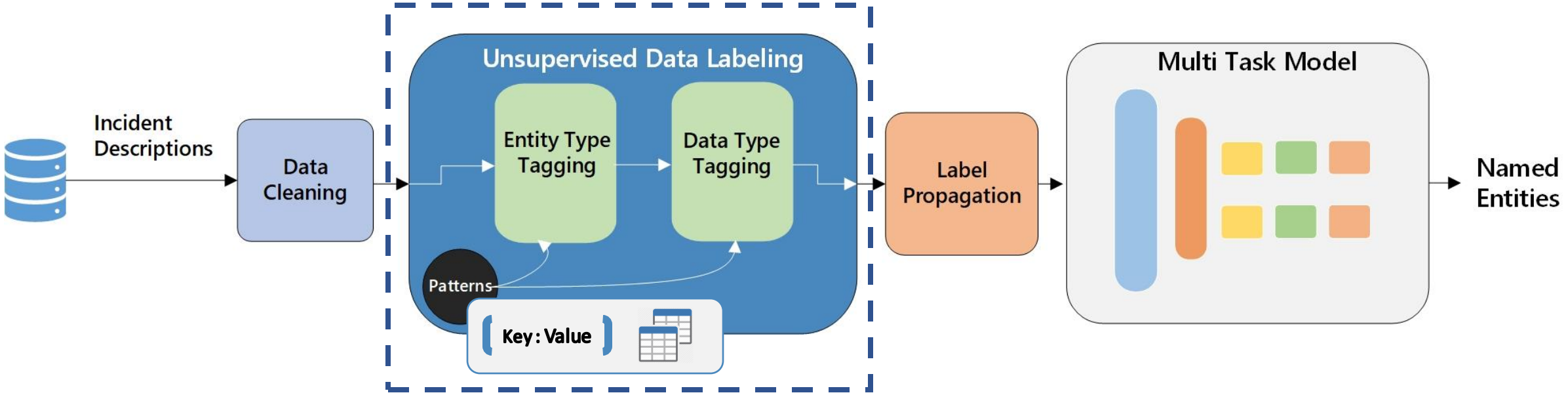
*Soft*NER: Machine Learning Pipeline

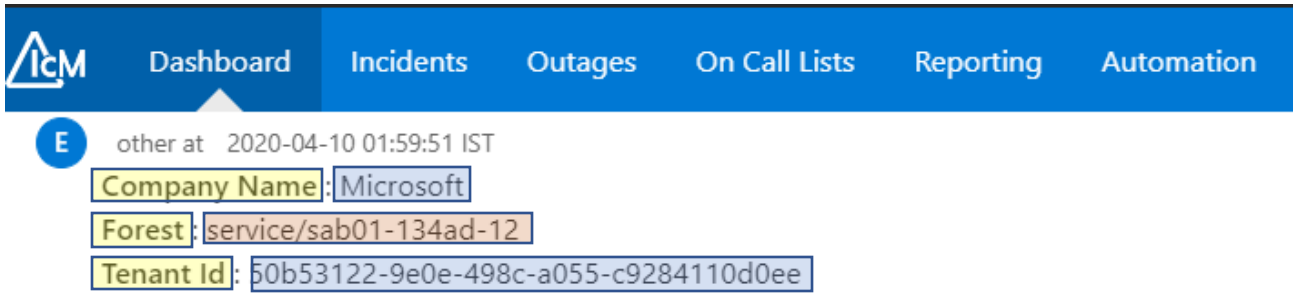


*Soft*NER: Machine Learning Pipeline

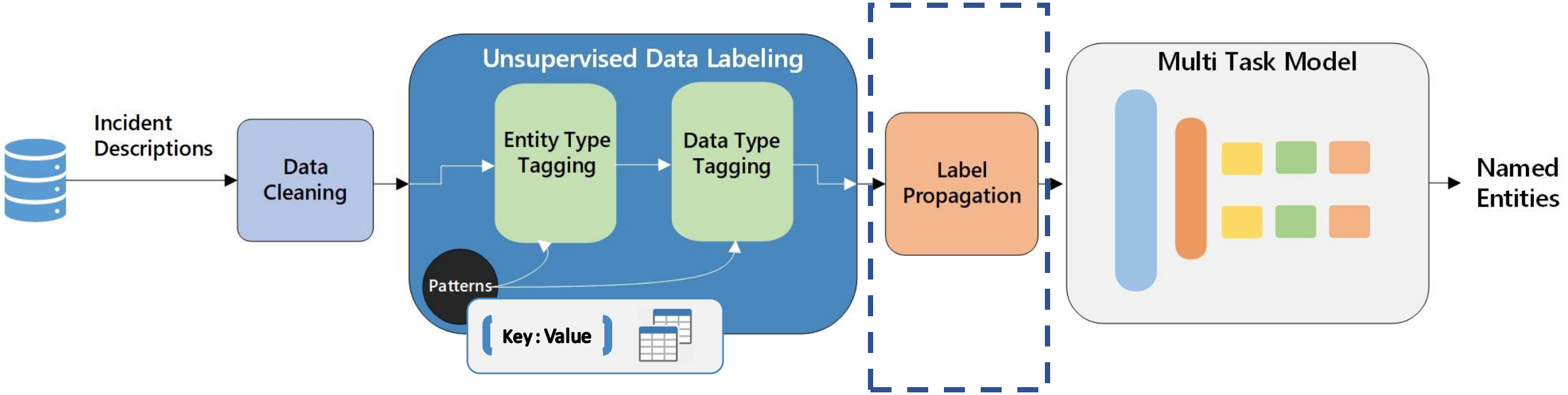


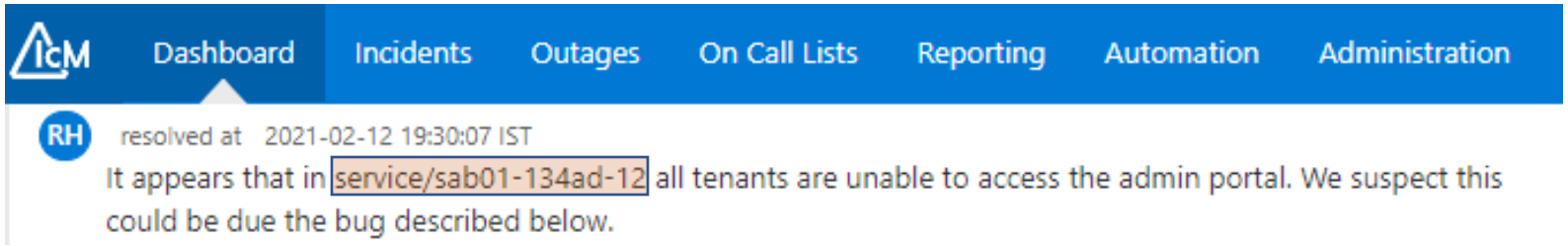
*Soft*NER: Machine Learning Pipeline



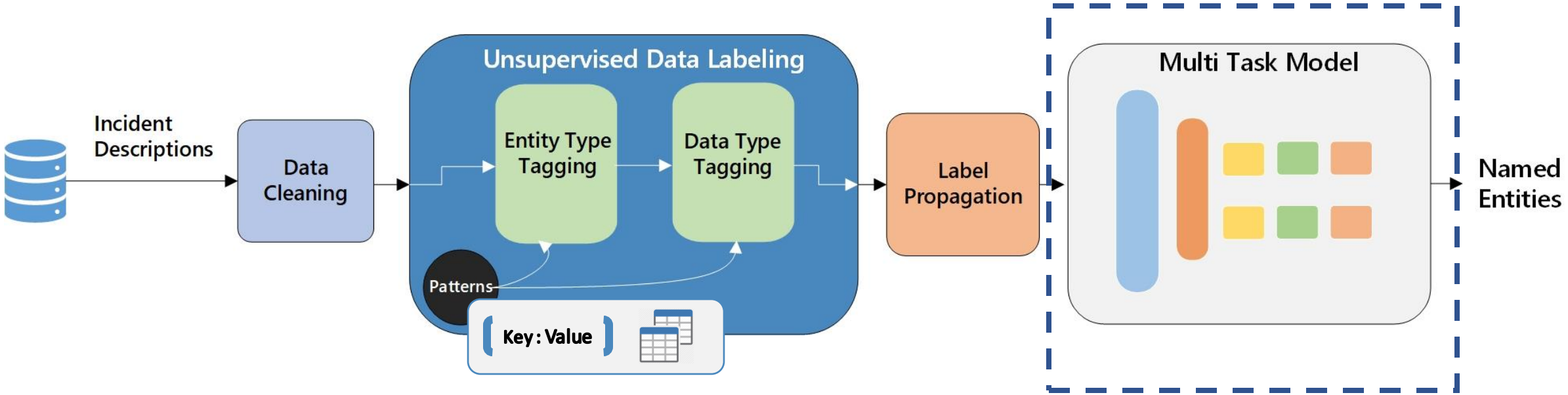


*Soft*NER: Machine Learning Pipeline





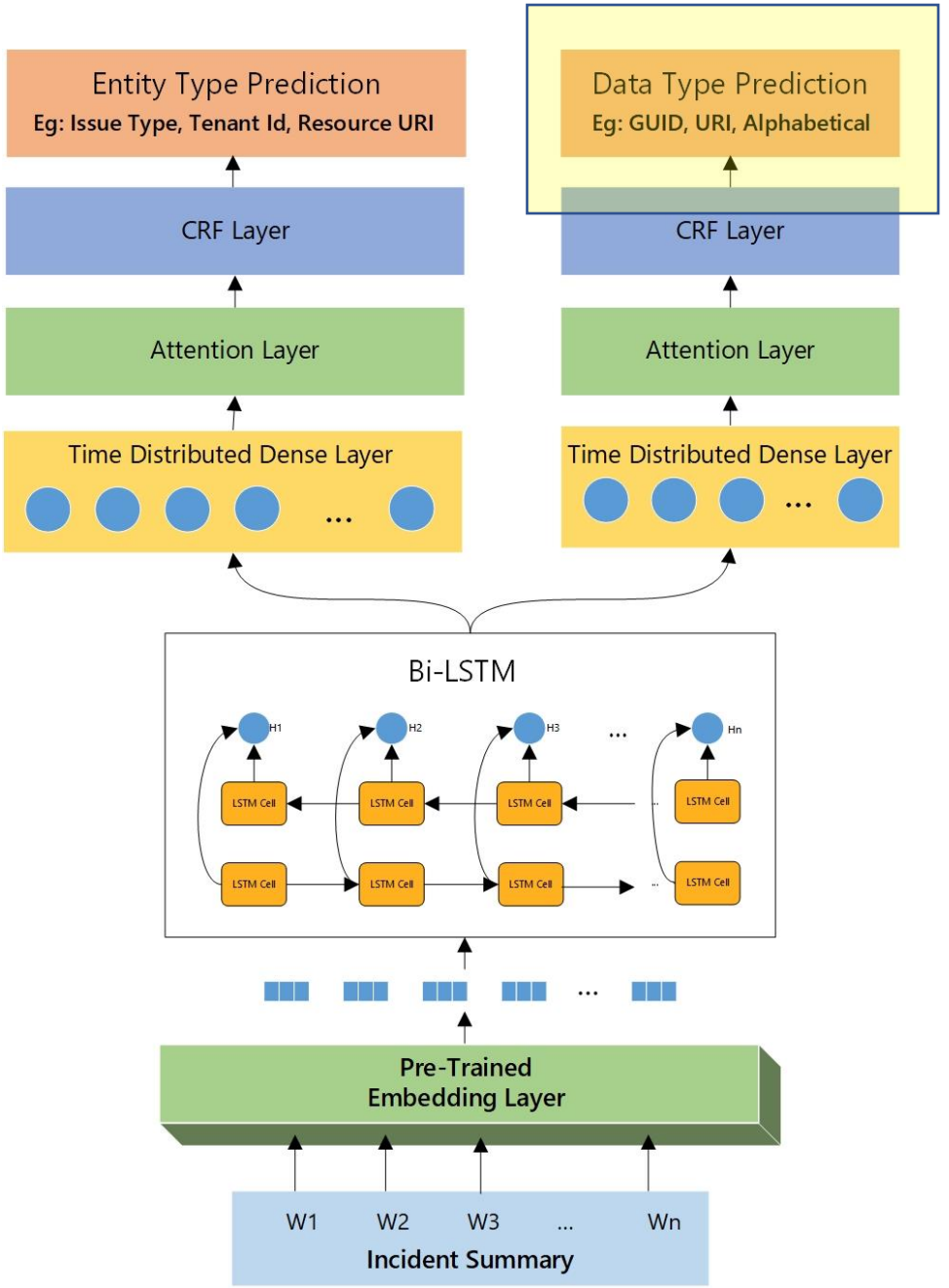
*Soft*NER: Machine Learning Pipeline



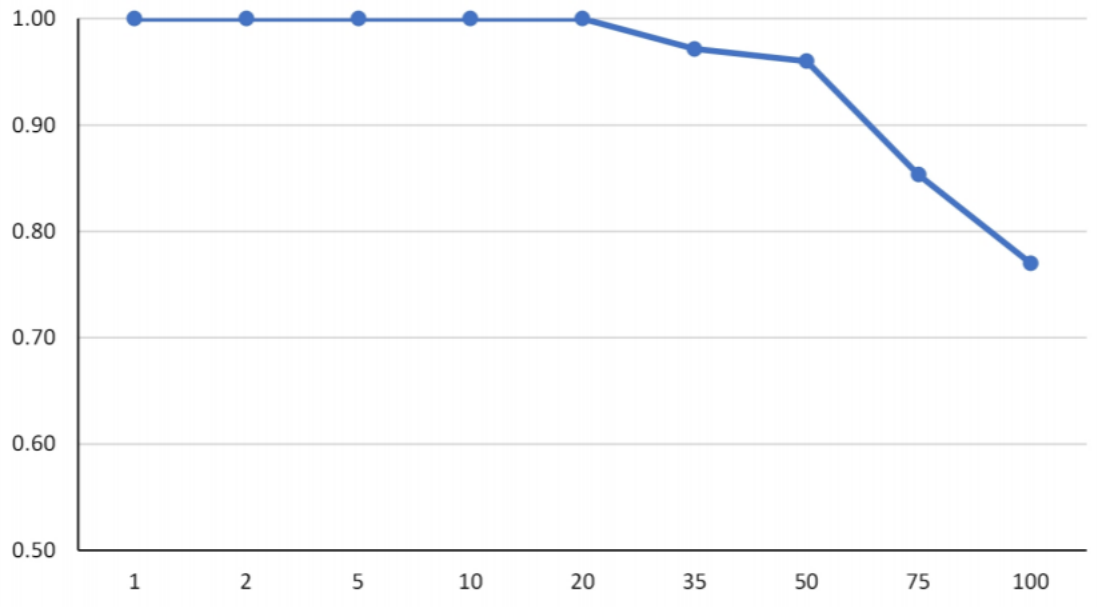
Model Architecture

Multi-Task model

with Attention Mechanism



Entity Type Evaluation



Precision vs Rank curve for the entity types

Model Evaluation

Metric	BiLSTM-CRF	BiLSTM-CRF Attention	SoftNER Model
Avg F1	0.8803	0.8822	**0.9572**
Weighted Avg F1	0.9401	0.9440	**0.9682**
Avg Precision	0.9160	0.9088	**0.9693**
Avg Recall	0.8669	0.8764	**0.9525**

Downstream Task: Incident Triage

Feature Set	Random Forest	Linear SVM	Gaussian SVM	K-Nearest Neighbors	Naive Bayes
Title + Summary	74.64	85.93	87.06	81.32	69.69
SoftNER Entities	93.38	93.34	93.39	92.40	87.67
Δ %	22.31	8.26	7.02	12.76	22.85
SoftNER Entities + Title	**98.60**	**99.20**	**98.95**	**99.14**	**88.07**
Δ %	27.66	14.34	12.78	19.75	23.30

Using the extracted entities for incident triaging

Future Work

- Expand to other DevOps artifacts like Troubleshooting Guides.
- Build and enable automated health checks that consume extracted entities.
- Transfer knowledge extracted by SoftNER to improve incident reporting tools.
- Enrich results by relation extraction and entity linking.

Summary

- DevOps artifacts like Incident reports and Customer tickets are unstructured.
- We have built SoftNER, a framework for knowledge extraction from Incidents.
- Multi-task BiLSTM-CRF model with an average F1 score of ~0.96.
- Integrated into the Incident Management platform @ Microsoft.
- Working on expanding SoftNER to other DevOps artifacts.