


Intent-Based Network Policy to Solution Architecting Recommendations

Aun Yichiet, Universiti Tunku Abdul Rahman, Malaysia

Jasmina Khaw Yen Min, Universiti Tunku Abdul Rahman, Malaysia

Gan Ming Lee, Universiti Tunku Abdul Rahman, Malaysia

 <https://orcid.org/0000-0002-0993-1130>

Low Jun Sheng, Universiti Tunku Abdul Rahman, Malaysia

ABSTRACT

The semantic diversity of policies written by people with different IT literacy to achieve certain network security or performance goals created ambiguity to otherwise straightforward solution implementations. In this project, an intent-aware solution recommender is designed to decode semantic cues in network policies written by various demographics for robust solution recommendations. A novel policy analyzer is designed to extract the intrinsic networking intents from ICT policies to provide context-specific solution recommendations. A custom network-aware intent recognizer is trained on a small keywords-to-intents dataset annotated by domain experts using NLP algorithms in AWS comprehend. The bin-of-words model is then used to classify sentences in the policies into predicted 'intent' class. A collaborative filtering recommendation system using crowd-sourced ground-truth is designed to suggest optimal architecting solutions to achieve the requirements outlined in ICT policies.

KEYWORDS

Intent Recognition, Machine Learning, Network Policy, Recommendations System, Solution Architecting

1. INTRODUCTION

Intent-Based Networking System (IBNS) integrates with machine learning to ease and automate some of the administrative works for network administrators and engineers. IBNS enables the conventional practices that require individual network-element configurations to be replaced by an abstraction layer that easily enables operators to express intent and subsequently validate the network to do what they desired to perform on the network. Unlike traditional tab completions found in some modern command line interface(cli); IBNS understand intentions context and can provide tailored solution recommendations when deployed at solution architectural level (Han, Li, Hoang, Yoo, & Hong, 2016). This simplifies the dev/ops of network solution architecting by improving implementation decisions, agility and fostering security with advanced automation.

DOI: 10.4018/IJBDCN.2021010104

This article, originally published under IGI Global's copyright on January 1, 2021 will proceed with publication as an Open Access article starting on April 1, 2024 in the gold Open Access journal, International Journal of Business Data Communications and Networking (IJBDCN) (converted to gold Open Access January 1, 2023) and will be distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

Solution architecting pipeline starts with high level policy making; usually from managerial people to solution architects to design a holistic network topology and finally to system administrators or network engineers for implementational works (Charalambides, et al., 2005). It is clear this is a complex pipeline that involves people with diverse IT literacy who are dealing with different constraints. Network policies are penned at a high abstraction level to account for business goals that benefits the organisation globally; which sometimes do not consider technical challenges and implementation feasibility to align with cost and performance constraints (Lupu & Sloman, 1999). The translation of intent to policies is a loosely process; whereby written policies in some cases do not reflect the original intents; especially when the gap of technical proficiency is profound. On technical fronts, another layer of loosely process happens when practitioners work to architect robust solutions to deploy the goals stated in the layman styled policies. This is an optimisation problem, where the objective function is to achieve high level goals using services, tools and technology that are attainable, scalable and manageable that are also at the same time, fitting to existing infrastructure. Optimal solutions are often case specific thus not easily carbon copied from some best practices architecture (Siau, Nah, & Teng, 2002). At the backend, there are ambiguous methods to measure optimality; whether it is latency, throughput or availability. At this stage, the experience of technical personnel in handling multi vendor solutions like ensuring interoperability between across Huawei, Cisco, 3com, Ericsson equipments might determine the robustness of resulting architecture. In addition, deep optimisation like conflict detection or synergy check among possible solutions often triumph deploying out of box solutions (Marcon, Dischinger, & Gummadi, 2011).

These constraints are recently addressed with Policy-based Network Management (PBNM). PBNM can be integrated with SDN, to delegate the entire management can encapsulate within a central unit that is smart enough to understand intents in policies and recommend contextual aware solutions (Avolio, Fallin, & Pinzon, 2007). In PBNM, the atomic unit is called *policy*. Policy, defined as the combination of rules and services where rules defined the criteria for resource access and usage. Each of the policies are composed a set of conditions and corresponding rules to overcome it. If the conditions of the policy rule are met, one or more actions contained by that policy rule may be executed. (Ding 2010). Policy-based management has simplified the complex task of managing networks and distributed systems. It could reduce the complexity of managing a large computer system and the manager does not need to constantly monitor the equipment and systems. PBNM allows admin to define various kinds of traffic metric such as data, voice, and video and assigns their priority of availability and bandwidth based on policy statements. The framework for PBNM over SDN use Neural Networks to monitor network parameters will adaptively react upon the policy violation. The use of the OpenFlow protocol enables the controller to communicate with the network devices to implement these changes. Meanwhile, ViVoNet (Chaudhari, et al., 2019) was developed by Django that use voice assistance to capture networking intents. Leveraging on NLP, the components of ViVoNet are runs as VMs on VMWARE ESXi hypervisor are trained to accept high level voice input for configuration jobs. The novelty comes from ease of configuration; but ViVoNet works on some pre-trained intents instead of some general purpose intents that are normally found on written network policies.

In IBNS domain, there are two general classifications based on the level semantic understanding of intentions found in network policies. In **general-purpose** system uses an average weighted score of network entities to recognise underlying intents. In a network policy, multiple intents or sub-intents sometimes co-exist and conflicting (Lupu & Sloman, 1999; Saha, Tandur, Haab, & Podieski, 2018). In this method, the algorithm simply takes the weighted sum of occurrence; or using term frequency to recognize the dominant intent among a set of intents. This is useful to remove some less important intents that are not directly related to networking, or could have been taken care of when the main intent is implemented. Meanwhile, **special purpose** system extract intents comprehensively at a finer level. Depending on contexts, special purpose recognition is needed for robust solution recommendations to fulfil multiple constraints (Singh, Rishi, Awasthi, Srivastava, & Wadhwa, 2020). This method removes

the assumption of one size fit all solution, instead, it recommends a set of solutions and delegates fine controls to them for better coverage of achieving network policies. They are more computationally intensive and can sometimes ‘over’ recommend due to model overfitting especially if there are many noises in the written policies. In this paper, we proposed a variant of general-purpose network solution recommender to automatically suggest optimal networking solutions based on intentions semantic extracted from written network policies. A novel semantic recogniser that understand networking semantics is designed to minimise intention ambiguity and solution mismatched due to the case of ‘loss in translation’ from policy to implementation. The paper is organised as follows: in section II, we discuss some related works in IBNS; section III outlines the design of Intent-Recognition-to-Solution-Recommendation Pipeline (IRSR); section IV evaluates the proposed method; and in section V we conclude the research findings and outline some future works.

2. RELATED WORKS

In IBNS, the perceptual gaps between administrative and technicality can be bridged using semantic aware systems. NLP is currently matured to takes on sophisticated data mining tasks to find, extrapolate and understand data points. Modern NLP model trained on deep learning is customisable, to be tailored for other domains (Rajani & Hanumanthappa, 2016). For example, a custom semantic model for computer network can be trained to accept high level queries and then extracting intents from network policies written in different levels of technicality (Mallamma & Hanumanthappa, 2014). Among many goals, two standout goals are in IBNS is mostly to reduce ambiguity in policies written by non-techie in human to human communication. IBNS is also important for engineers to communicate with devices using intuitive interfaces and languages for streamlined and simplified configurations (Mihaila, Balan, Curpen, & Sandu, 2017). IBNS can be delivered as an integration onto SDN or as simple as adding intent-aware tab completion functionalities to Cisco IOS CLI. Comparatively, Cisco IOS currently use naive ‘unique character filtering’ to narrow down command options but does not intuitively recommend suitable command options based on contexts.

2.1 Policy Ambiguity

As network environment are getting complex, policy-based management facilitates policies provisioning to achieved the desired quality of service (QoS). However, conflicts of policy specification may lead to policy ambiguity and unpredictable effect on network performance. When there are multiple similar policies coexist, there is a possibility that policy will be in conflict either because of specification error or application-specific constraints (Lupu & Sloman, 1999; Charalambides, et al., 2005). Modality conflicts is one of the main causes of policy ambiguity. If two of the policies are specified using the same subjects, targets, and actions but different set of policies applied. It may lead to overlap between the subjects, targets, and actions of the policies (Lupu & Sloman, 1999). For example, when a network administrator wants to restrict only 1GB of the Internet bandwidth for every user within an organization network, he set the policy as the first network policy. However, the director is excluded from that policy and he has the privileges to used up to 5GB of Internet usage within the organization network. Hence, the network administrator once again set the policy that allow the director to used up to 5GB of Internet usage. Here is a conflict between the first and second policy as network administrator already mentioned at first all the user only can used up to 1GB which include the director as well, hence the second policy will not work even it is specified. To resolve this issue, network administrator may consider resolving the policy conflicts by assigning precedence to policies. By establishing policy precedence, it allows two similar policies coexist within the network system and determine which policy should be applied. Policy conflicts could be substantially reduced by establishing the precedence and priority between different policies (Chalon, Durand, & Richard, 2001; Yao, Jiang, & Qian, 2019). Let take the previous example, if the network administrator changes the arrangement of the policies by setting the director having 5GB of Internet

usage as the first policies, and the second policy set all other users only can used up to 1GB of Internet usage. There will be no conflict between these two policies as the first policy will have the priority over the second policy (Nair & Novak, 2007; Charalambides, et al., 2005).

Two important nomenclatures in IBNS can be defined into '*intent*' or '*goal*'. Intent and goal have a strong relationship in understanding a policy and giving out recommendations. First of all, intention is a general term that try to bring up actions around, and goal is a more concrete actions that want to accomplish (Azeem & Sharma, 2017; Kaviani, 2017). When understanding a policy, we may first want to extract the intent as intent bringing up actions. For example:

- **Intent:** intents are abstract, intrinsic and not directly observed but leads towards some common goals. Consider a network administrator who wishes to improve the security of the organization network. Hence, '*improves security*' is the intent of the text as it is the objective of the text and could bring more actions such as installing firewall, setting ACL or setting VLAN (Han, Li, Hoang, Yoo, & Hong, 2016; Sanvito, et al., 2018; Saha, Tandur, Haab, & Podieski, 2018);
- **Goal:** a network administrator wishes to improve the security of the organization network. Hence, '*improves security*' is the intent of the text as it is the objective of the text and could bring more actions such as installing firewall, setting ACL or setting VLAN (McSweeney, 2019).

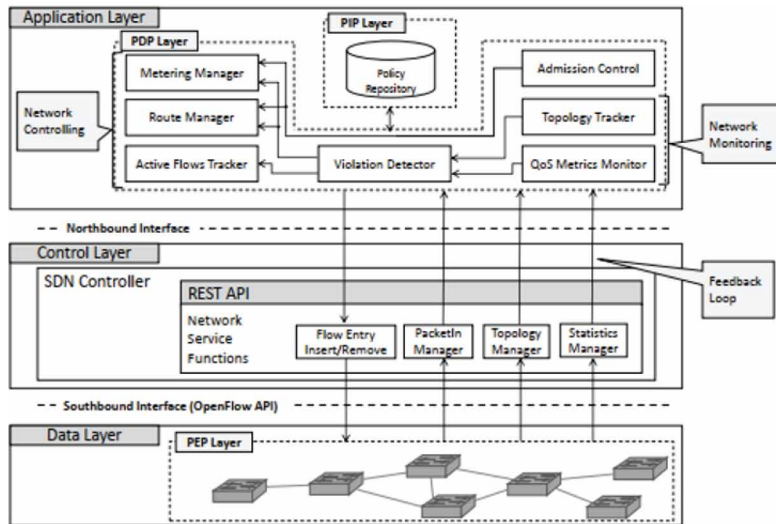
The relationship between intent and goal is strongly correlated, the goals may come with an intention and an intention may bring up goals. In computing, semantic analysis could be one of the ways to understand the information of a text and figuring out what the actual intent was based on the context in the text (Foltz, 1996) and setting up goals will be based on the result from semantic analysis and giving out related recommendations to achieved the intention of a text. One of the possible methods for the system giving out goals or suggestion could be embedded neural network in recommendation system (Singh, Rishi, Awasthi, Srivastava, & Wadhwa, 2020).

2.2 Policy Implementation Using Software-Defined Networks

Policy defined as the combination of rules and services where rules defined the criteria for resource access and usage. Each of the policy composed a set of conditions and corresponding rule to overcome it. If the conditions of the policy rule are meet, one or more actions contained by that policy rule may be executed (Lupu & Sloman, 1999). The framework for PBNM over SDN which monitors network parameters will adaptively reacts upon the policy violation. The use of OpenFlow protocol enable the controller to communicate with the switches. This work uses Neural Networks to identify the violating flow that deteriorate the network performance. Upon the detection of policy violation, route management such as rerouting and rate limiting will be implemented (Wickboldt, Jesus, & Isolani, 2015; Latah & Toker, 2016).

PBNM based on SDN framework consist of Northbound interface and Southbound interface. It consists a number of modules and each of the modules in the framework consist of different functionalities (Latah & Toker, 2016). For example, policy repository stores all the high-level policy which reflecting the requirements of pre-configured customer services and violation detector will validate and release the necessary measures to merge with the state agreed by SLO requirements. The functional components are mapped to three-level PBNM framework which are Policy Information Point (PIP) layer, Policy Decision Point (PDP) layer, and Policy Enforcement Point (PEP) layer. Upon receiving a new route request, the controller read the packet-in message and the admission control decides whether to accept or reject the upcoming request based on the availability of the resources. If the request is accepted, the controller will determine the best shortest path by using Dijkstra's algorithm. In PBNM, the network monitoring components periodically collect flow statistic from the switches in order to calculate the bandwidth available and the packet loss rate. If a high-level policy rule is violated, the violation detector will identify the flow that causes the violation by comparing the calculated quality metrics with the predefined high-level policy. A neural network will be used

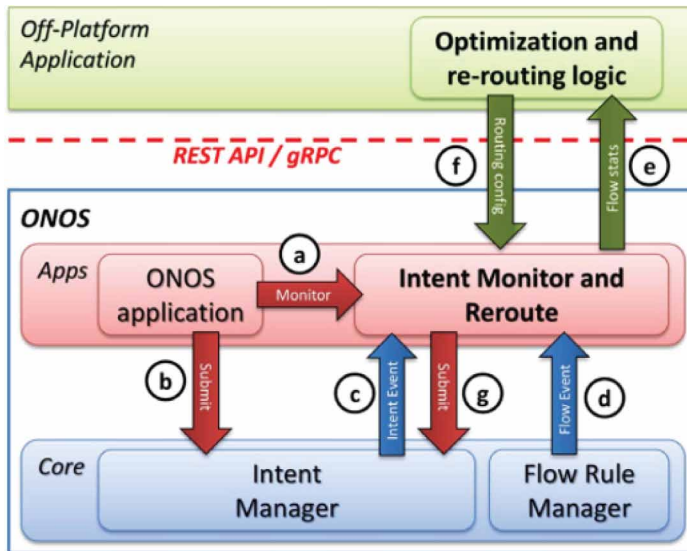
Figure 1. PBNM-based SDN framework



to identify the violating flows that cause network congestion. After that, the violation detector can choose to either invoked the route manager choose another best route based on Dijkstra's algorithm or reduce the bandwidth budget for the best-effort flows. PBNM-SDN users perceive a better quality with rerouting of network where the efficiency has increased 94%. PBDN-SDN is also highly scalable towards network growth and policy changes over time.

An Open Network Operating System (ONOS) application developed by Open Networking Foundation (ONF) allows users to specify their intent without have to worry any low-level functionalities (Imran, 2017). It designed to meet the needs of operators to create and deploy new dynamic network services with simplified programmatic interfaces. As ONOS supports both configuration and real-time control of the network, it eliminates the need to run routing and switching control protocols inside the network fabric. In ONOS platform, it consists of a set of applications that act as an extensible, modular, distributed SDN controller. And it is a scale-out architecture to provide the resiliency and scalability required to meet the rigors of production carrier environments. An ONOS service can carry out by monitoring and rerouting of the intents and defining a set of Application Programming Interface (API) to make it communicate with an Off-Platform Application (OPA). The OPA will implement the re-routing logic which range from classical optimization tools to Machine Learning or Artificial Intelligence approaches based on collected statistics. There is a service which called Intent Monitor and Reroute (IMR) which allows ONOS applications and users to specify a set of intents whose statistics are monitored and exposed to an external routing logic (Imran, 2017). The function of IMR is for interacting with ONOS Intent Manager and Flow Rule Manager to keep track of the mapping between the intent and the corresponding flow rules and their information. Intent Monitor and Reroute (IMR) was used to offer ONOS applications and users desire to monitor and re-routing of specific intent. ONOS monitoring will periodically retrieving statistic of each low-level flow generated and apply corresponding intent in the network. IMR always have to interacts with ONOS Intent Manager and Flow Rule Manager to keep track of the mapping between the intent and corresponding flow rules and other related information. IMR have 3 states for an intent which are Not Monitored, To Be Monitored and Monitored. By default, the intent state is Not Monitor until when a user or application require to monitor an intent, the state will either changes to To Be Monitored or Monitored. If the state is in Monitored, it will start tracking of its statistics (Imran, 2017; Sameer, 2015). Based on the statistics retrieved from IMR, Off-Platform Application (OPA) can specify particular

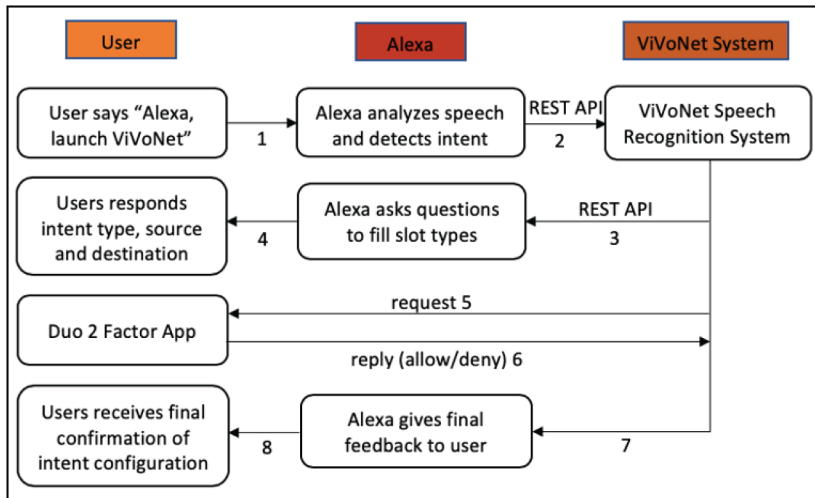
Figure 2. IMR with ONOS and with the OPA



path for each of the intent in order to optimize the network performance. Through this mechanism, IMR just have to submit the intent keys and the rest of the work will be left to OPA. OPA will collect the statistics, optimize the algorithms and reroute the intents if necessary. Different network objectives will achieve with different routing logics by the OPA, Clustered Robust Routing (CRR) is an adaptive Robust Traffic model that allow to tune the trade-off between dynamic routing and stable routing. The historical data will be feed into the optimization model over the training period. The model will compute a series of routing configurations that will be applied in the network. To deal with dynamic traffic, routings are trained with more robust over subsets of traffic matrix space. With ONOS, the monitoring executed by OPA is important to handle any unexpected traffic scenarios. For each of the ONOS module, it can optimize the routing logic in ONOS applications based on intents, via an external plug and play routing logic with a few modifications to the application code (Sameer, 2015). The performance can be easily improved by leveraging routing logic decoupled from the application which requested the intents.

VivoNet is an existing policy that automate network configurations through intent; specifically, voice commands. The system will accept voice inputs from user and dynamically convert it into intents and implement on suitable network infrastructure (Chaudhari, et al., 2019). The front end and the REST API were developed by Django. The components of VIVoNet are virtualized on virtual machine on VMWARE ESXi hypervisor. The virtual switches used in this development is Open vSwitch(OvS) that running OpenFlow 1.3 and there is a centralised SDN controller to manage all the OvS. The development of the project is believed have a significance impact to network administrators as well as visually-impaired audience. Users talk directly to Amazon-Alexa service Alexa to activate custom skill has been created by VIVoNet using Alexa Skills Kit (ASK). The custom skill collects a set of sample utterances to invoke specific intents and slot types to stimulate the input from user. In the VIVoNet system, the intent Engine responsible for converting user-requested intents into translated network configuration. The Engine will create appropriate flows and the controller will pushes them out using OpenFlow protocol to the switches. The slot-types values and the intent will be sent to REST API by POST request. VIVoNet can achieve agility as it required 0.0016 seconds on average to create, push and verify flow for one intent (Chaudhari, et al., 2019).

Figure 3. The workflow for Alexa



2.3 Natural Processing System (NLP) for Semantic Analysis

In NLP, there are four types of morphological processing: Lexical Analysis, Syntax Analysis (parsing), Semantic Analysis and Pragmatic Analysis. Semantic analysis gives the exact meaning or directory meaning from structures created by syntactic analysis. To capture the real meaning of an sentence, it will first identify the text elements and assigns them to their logical and grammatical group. After that, it will structure each element and identify the meaning of the text. For example, the terms economic can be related to networking, security, performance or Artificial Intelligence. There are a few implementations of semantic analysis in real life application. (Foltz, 1996) proposed a paper for text comprehension of the semantic similarity between different pieces of textual information using Latent Semantic Analysis (LSA). In text-comprehension research, subject is read from textual information and provide some form of summary. The summary documents are used for the researcher to identify what information the subject has extracted from the text. Researches examine each of the sentence in the subject summary and match the information contained in the original sentence in the text. To analyse a text, LSA is used to generate a matrix of occurrence of each word in a sentence or a paragraph. After that, it used SVD (Singular Value Decomposition) to decomposes the word-by-document matrix into a set of k of orthogonal indexing dimensions. The author has made an experiment by taking a text and writing them into summary (Foltz, 1996). After that, it will match the individual sentences from the text and match each individual sentence from the subject's summary. As result, the summary is highly semantically similar to original texts. Meanwhile, (Mallamma & Hanumanthappa, 2014) by using semantic tools for extracting information to from unstructured data and present it to the user through spring graph. As nowadays large volume of information spread across the web and it will become useless if we can't locate and extract the correct information from unstructured data. Hence, the researchers came out with an idea by analysing data through semantic tools. They will first structure the data by providing a set of entities within a domain such as company, person, location and organization. Based on the information they extract for relevant entities; the result will be represented into RDF (Resource Description Framework) graphs.

For extracting a structured data from an unstructured data, the authors have used Ontology learning method to define the concept and associations by extracting domain terms, concepts, concept attributes and relations from textual data (Mallamma & Hanumanthappa, 2014). Ontology learning primarily focused on defining the concepts and associations between them. It extracting domain terms, concepts, individuals, concept attributes and relations from textual data. To represent

Figure 4. Example of RDF description graph

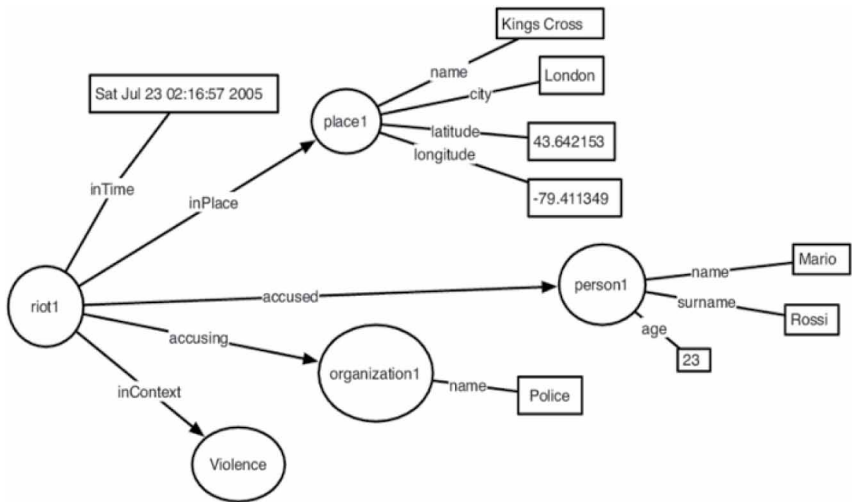
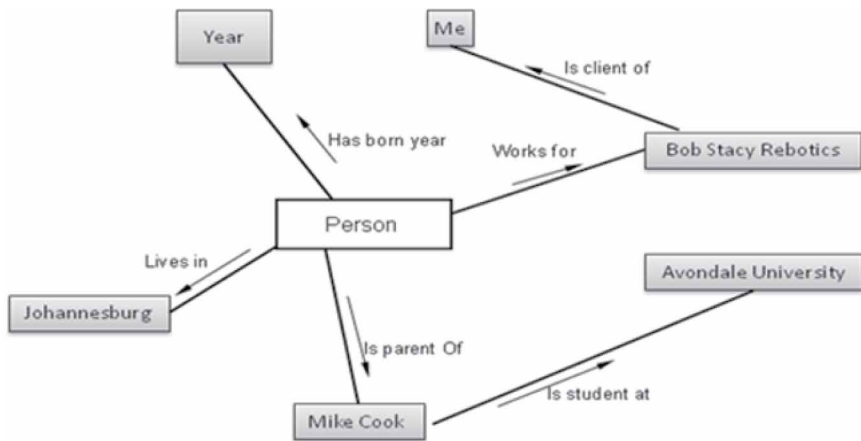


Figure 5. Example of RDF ontology

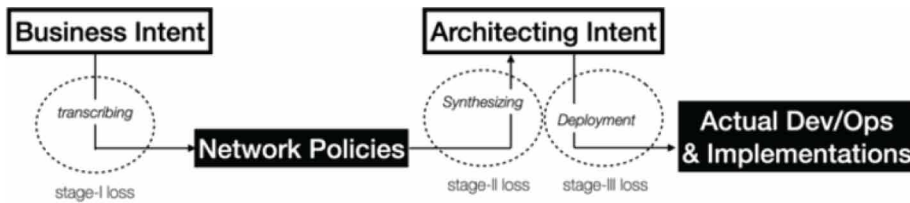


ontology extraction in graphical format, the data will be visualized in spring graph and it will be used in information extraction from a digital library after semantic analysis.

To organize information from an unstructured data, (Rajani & Hanumanthappa, 2016) used domain ontology for semantic analysis of natural language queries. Ontology is a types, properties, and interrelationship of the entities. All the natural language queries will be sent for Natural Language Interface to Database (NLIDB). NLIDB used to take structured information from database and compare the request by using natural language and after that submit his or her query to database. This means any queries will be pre-process by using semantic analysis before sending to the database. From the queries, ontological semantic provides the language independent, meaning based representations of concept. A database query translated from NLIDB will eventually bring the preferred information from a query database to the user.

Based on the application as stated above, it can prove that to process any natural language, semantic analysis is necessary. Without the syntax and semantic analysis in the machine translation,

Figure 6. The case of 'loss in translation' of original intents from penning the policies to understanding it and actually implementing it



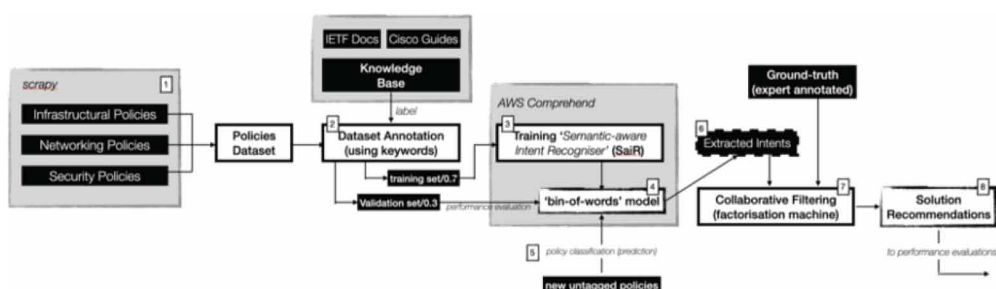
the result may be ambiguous and the machine could not understand the natural language (Desot, Portet, & Vacher, 2019; Serdyuk, et al., 2018).

3. INTENT-RECOGNITION TO SOLUTIONS RECOMMENDATION PIPELINE (IRSR)

Network solution architecting is a complex pipeline in aligning business goals to network policies while considering implementational constraints. The challenge is to find the most fitting network solutions to achieve policies in accordance to certain business compliances, like cost and security. We model this as an optimisation problem, where the objective function is to identify a set of optimal solutions to execute written policies while navigating through implementational constraints. Our intuition is that information that is loss in translation as a significant factor on why network policies and their corresponding solutions are often mismatched. Specifically in a multi-literacy context, network policies can be written by business minded people in layman language and executed by technical engineers. In a lossy translation, the network policies might not convey the original business' intentions; and the execution might not reflect the policies intentions. There are two stages of loss from defining policies to actually implementing them: (1) when business leaders define certain tangible goals in vaguely written policies (they know what they want but not sure about their options) and (2) when the written policies are being comprehend by solution architect (they know their options but are not exactly sure what's the goal to achieve) and (3) the goal is understood but practitioners are confused with their options. Figure 6 visualises the three hotspots of loosely information translation from policy to solution.

We developed IRSR to delegate 'understanding' network policies to machine-learning; leveraging on semantic engine in NLP domain. IRSR starts with a Semantic-Aware Intent Recogniser (SaiR) that is a parser designed to identify, capture and extract networking intentions from policies in textual format. Meanwhile, an Intention-to-Solution recommender (ISR) is designed to provide network solutions recommendations that aligns with intents previously captured in SaiR based on crowd-sourced consensus for automated solution architecting. Figure 7 visualises the overview of IRSR.

Figure 7. The overview and components of the proposed intention-aware solution recommendation



3.1 Semantic-Aware Intent Recogniser (SaiR)

The goal of SaiR is to actively ‘look’ for networking intentions in any written network policies. SaiR framework is built on a collection of AWS services; including AWS Lambda, Comprehend, DynamoDB and S3. Technically, SaiR is keyword recognition model that is trained using networking semantics to detect for overt and non-overt networking intentions. Figure 8 shows the components in SaiR. The inputs to SaiR are alike to classic machine-learning (ML) architecture but we only deals with text data with no accompanying visuals like network topology diagrams. The dataset is crawled from public webs using a custom web-crawler built on scrapy (see Figure 9) to crawls relevant data like network policies, ICT strategies and architectural blueprints. To prevent potential model overfitting, we collect diverse policies with different technical proficiency, scope and coverage from webs and companies with different calibre. Figure 10 shows a sample compilation of 4 different policies.

3.2 Data Preparation

Policies scrapped from the net is first normalised to remove bad data points with a domain analyser. We built domain classifier into Scrapy to filter out non-technical texts like HR and operational policies. This ‘noise-filter’ reduce articles ratio from 30:1; note that we are not concerned on wrongly discard relevant policies but instead concerns on allowing irrelevant policies through; provided that we have enough samples for our training. Sentences in these selected policies form the atomic unit for SaiR; since SaiR analyses policies by individual sentences. We use ‘dot’ as the delimiter to break

Figure 8. The components in SaiR; starting with data preparation to model training and intent classification

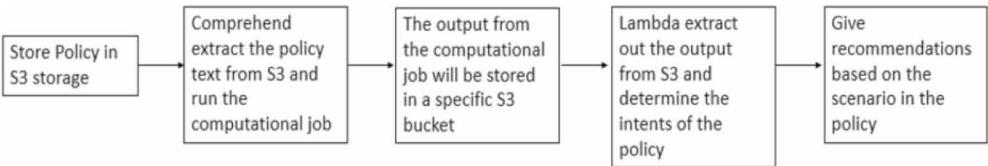


Figure 9. The web crawler’s design to automatically collect network policies from public web

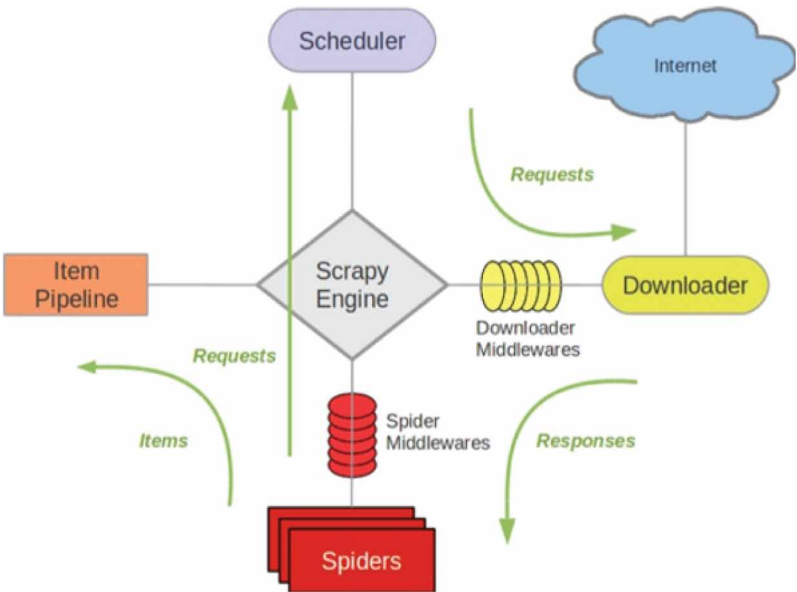


Figure 10. A compilation of policies with different attributes: (a) Generic policy; (b) Technically sounded policy; (c) Policy for a specific University's faculty; and (d) Policy of a Forbes Top 100 company

<p>Users are prohibited sending or displaying intimidating, hostile, offensive, or hate related material. They are also disallowed to use obscene language or intentionally accessing or possessing obscene or pornographic material.</p> <p>General Policy</p>	<p>Any web content which related to fraudulent, harassing, obscene (i.e., pornographic), or threatening material should be blocked within the Local Area Network</p> <p>Technical Policy</p>
<p>Users must not send, view or download fraudulent, harassing, obscene (i.e., pornographic), threatening, or other messages or material that are a violation of applicable law or University policy.</p> <p>University Policy</p>	<p>The equipment, services and technology used to access the Internet are the property of company and the company reserves the right to monitor Internet traffic and monitor and access data that is composed, sent or received through its online connections</p> <p>Tech Company's Policy</p>

paragraphs into sentences. Each sentence denotes one instance in the text 'corpus' which are stored on AWS DynamoDB.

The dataset is labeled using an aggregated domain expertises for subsequent weakly supervised learning. A set of keywords are derived from 48 sources that includes Cisco's networking guides, IETF documentations and prominent technical papers (see Table 1). We derived a custom networking ontology based on the consensus of these sources to automate labeling intents in the sentences; whereby each sentences will be categorised as one of five classes, namely: *{Authentication, Network_security, Performance, Access_control, and Self_healing}*. For example, words like 'password', 'identity' and 'username' are discriminators for the 'authentication' class; while words like 'quota', 'monitoring', 'obscene' pertains to the 'access_control' class. For sentence with conflicting keywords, we use term-frequency (TF) to selectively label the sentence with the dominant class. Using domain knowledge, the label generator predicts and label each sentences with highest weighted *intent* class. We extrapolates the dataset using two simple data augmentation techniques: (1) inverting sentences structure from active to passive, vice versa; and (2) manually add sentences as padding so all classes have same number of samples to prevent data imbalance issue. Figure 11 shows the truncated examples of the labeled dataset.

3.3 Model Training

SaiR leverages on NLP algorithms in Amazon Comprehend to extract key phrases, entities, and sentiments automatically. The model is trained using sentences that are labeled with intents based on per-sentence dominant keywords. We opt for AWS comprehend to delegate ML technicalities to AWS

Table 1. Sample utterance to invoke intents

Slots Name	Slots Type	Sample Slots Values
{intent_type}	intent_type	Least hopcount, least latency, high bandwidth
{from_city}	AMAZON.City	Denver, San Francisco (any U.S. city)
{to_city}	AMAZON.City	New York, Chicago (any U.S. city)
{confirmation}	confirmation	Yes/No

Table 2. A snippets of technical keywords to identify networking ‘intents’ as aggregated from Cisco guides, IETF documentations and technical papers. The example is pruned for clarity of visualisation.

domain controller	AUTHENTICATION	(Rouse 2006), (authorization, n.d.)
authentication server	AUTHENTICATION	(Poremba 2017), (Rouse 2018).
username and password	AUTHENTICATION	(Rouse 2006), (authorization, n.d.)
User Identity Verification	AUTHENTICATION	(Authentication, Authorization, Cisco IOS Release ISSY 2019)
pornographic	WEB_ACCESS_CONTROL	(Stanford Computer and Network Usage Policy 2014)
monitored	WEB_ACCESS_CONTROL	(Credentials Processes in Windows Authentication 2016)
monitoring	WEB_ACCESS_CONTROL	(Internet And Email Access Policy 2015)

Figure 11. Showing samples of dataset labeled with networking ‘intentions’. Figure is truncated to fit examples from all classes.

PERFORMANCE	Staff offices are also monitored for bandwidth usage. The limit for office computers is typically one gigabyte of data per day (upload and download), although exceptions can be made.
PERFORMANCE	Bandwidth both within the company and in connecting to the Internet is a shared, finite resource. Users must make reasonable efforts to use this resource in ways that do not negatively impact the network.
PERFORMANCE	Network bandwidth is used when a person uploads or downloads data on the Internet. Network administrator will monitor bandwidth consumption to make sure that this shared resource is used efficiently.
PERFORMANCE	Download and upload speeds to the Internet are limited to 2 Mbps in order to provide a consistent quality of service for all devices on the network. Faculty and staff offices are also limited to 2 Mbps.
PERFORMANCE	Network bandwidth is the capacity of a wired or wireless network communications link to transmit the maximum amount of data from one point to another over a computer network.
AUTHENTICATION	Unless company staff have proper authorization, users must not attempt to modify or remove information or information resources that are owned or used by others.
AUTHENTICATION	Authorization is the function of specifying access rights/privileges to resources, which is related to information security and computer security in general and to access control in particular.
AUTHENTICATION	Authorization in computer systems and networks rely on access policies. The access control process can be divided into the following phases: policy definition phase and policy enforcement phase.
SELF_HEALING	Any interruption of network services can affect an organization's ability to access, collect or use data and communicate with staff, partners and customers. Interruption of network services can be caused by a variety of factors.
SELF_HEALING	Network equipment failure such as routers, switches, modems, gateways, or any other device can affect the performance of all other devices connected to them. It can be caused by a variety of factors.
SELF_HEALING	A single network consists of multiple routers, nodes, or switches. Failure of one network components might become overloaded and stop working, which can trigger a disaster of any type can significantly damage or even destroy your production center and virtual infrastructure, thus causing significant business losses. Any interruption of network services can be caused by a variety of factors.
SELF_HEALING	Data centers must be able to quickly recover from equipment failure. This is especially true for applications such as e-commerce where a significant amount of money is at stake.
SELF_HEALING	There are three principles of systems design in reliability engineering which can help achieve high availability. First, we need to eliminate single points of failure. This can be achieved by using redundant components.
SELF_HEALING	Systems that exhibit truly continuous availability are comparatively rare and higher priced, and most have carefully implemented speciality designs that eliminate any single points of failure.
NETWORK_SECURITY	Central campus network and security personnel must take immediate action to mitigate threats that have the potential to pose a serious risk to campus information system resources.
NETWORK_SECURITY	Unauthorized network access – An attacker needs access before being able to perform any attacks. An attacker can be a disgruntled employee, an employee that has become a former employee, or an external attacker.
NETWORK_SECURITY	Unauthorized network access is accessing your network with intrusion techniques. Whether it be to protect yourself from malware or ensure that your private information is secure, you need to protect your network.
NETWORK_SECURITY	A firewall is a network Security device that monitor incoming and outgoing network traffic and devices whether to allow or block specific traffic based on a defined set of Security rules.
WEB_ACCESS_CONTROL	Users must not send, view or download fraudulent, harassing, obscene (i.e., pornographic), threatening, or other messages or material that are a violation of applicable laws, regulations, or University policies.
WEB_ACCESS_CONTROL	Every employee is responsible for the content of all text, audio, video or image files that he or she places or sends over the company's Internet and e-mail systems.
WEB_ACCESS_CONTROL	Inspecting and monitoring information and information resources may be required for the purposes of enforcing this policy, conducting University investigations or audits, or for other purposes.
WEB_ACCESS_CONTROL	Company employees are expected to use the Internet responsibly and productively. Internet access is limited to job-related activities only and personal use is not permitted.

while ensuring end-to-end continuity of AWS services in the pipeline. The output is a multi-class classification model that understand networking semantics based on words composition in a sentence. In the training, SaiR learns the *keywords to intents* correlations based on expert labels, then attempts to categorise other words in the same sentences to that particular intent. Intuitively, it is easier to let the algorithm to figures out on *words to intents* correlations using *term-frequency (TF)* or *bags of words* but this requires a large training data. Instead, SaiR compute the relevancy of each words to the ‘intent’ classes and group them into respective ‘bins’ using the labeled keyword as ground truth. We use train-test ratio using 840 documents with arbitrary number of sentences for modeling. The exact ML configurations and para-meters tuning are abstracted by *Comprehend*. and Figure 12 illustrates the intuition of SaiR in modeling intents associated with sentences using constrained dataset.

Since SaiR is designed for policy-level prediction; SaiR first classifies sentences individually then aggregates them (per policy) to find a common class. Figure 13 visualises the logic of intent classification for each policy using the weighted sum of individual sentences’ class.

Using the classification model, we design lambda functions to automate intent predictions; whereby each words in every sentences are classified with a label. The prediction job is simply a json pass to the model; with accompanying metadata like {‘JobName’, ‘EntityRecognizerArn’, ‘LanguageCode’, ‘DataAccessRoleArn’, and ‘clientRequestToken’}. ‘JobName’ is the identifier

Figure 12. SaiR uses NLP algorithms (in Comprehend) to 'teach' the model on 'other possible words' that belongs to an intent class based on a few hand labeled keywords in a sentence. From top to middle, SaiR learnt that 'password' and 'authorize' belongs to 'intent:authentication'. From middle to bottom, SaiR learns that other words in the same sentence: 'users', 'configure', '2-factor', 'identity', 'IAM' possibly relates to 'intent_authentication' and they are placed into the same bin (of intent).

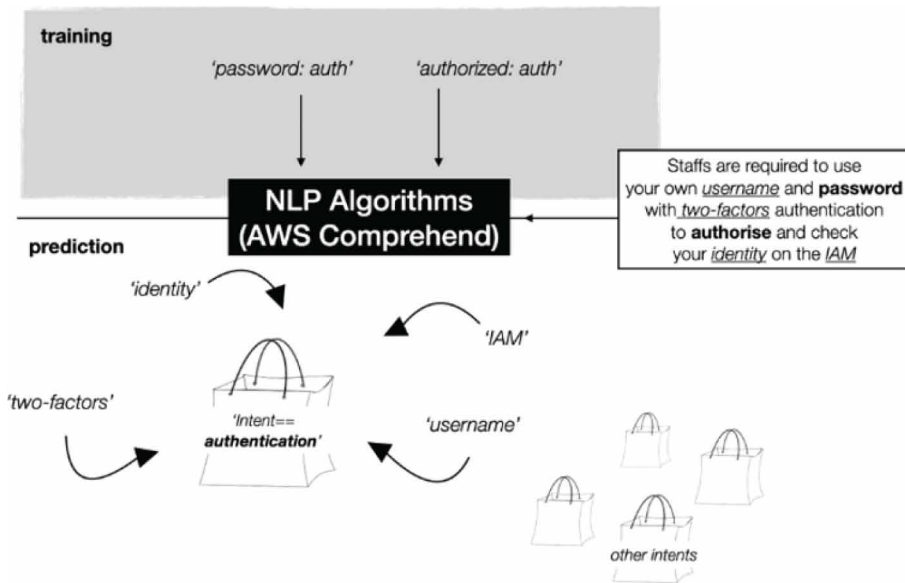


Figure 13. A prediction request for a text based policy which is classified as authentication based on weighted sums of class(es) for all sentences. The most dominant intent is the intent with largest weighted sum, denoted with the largest circle.

```
{
  "Entities": [
    {
      "BeginOffset": 38,
      "EndOffset": 48,
      "Score": 0.9999929666519165,
      "Text": "fraudulent",
      "Type": "WEB_ACCESS_CONTROL"
    },
    {
      "BeginOffset": 50,
      "EndOffset": 59,
      "Score": 0.9999924898147583,
      "Text": "harassing",
      "Type": "WEB_ACCESS_CONTROL"
    },
    {
      "BeginOffset": 61,
      "EndOffset": 68,
      "Score": 0.9999842643737793,
      "Text": "obscene",
      "Type": "WEB_ACCESS_CONTROL"
    },
    {
      "BeginOffset": 76,
      "EndOffset": 88,
      "Score": 0.9999842643737793,
      "Text": "pornographic",
      "Type": "WEB_ACCESS_CONTROL"
    },
    {
      "BeginOffset": 225,
      "EndOffset": 238,
      "Score": 0.9998327493667603,
      "Text": "authorization",
      "Type": "AUTHENTICATION"
    }
  ],
  "File": "NewlyCreated2.txt"
}
```

you give for the computational job. `'EntityRecognizarArn'` is an attribute that identifies the specific entity recogniser to be used by the method (here we point to SaiR model). For each of these words, a lambda will specify the begin offset and end offset of the words from the text and the precision of the keywords is properly labeled. Figure 13 shows an example of prediction results using some random network policy.

3.4 Intention-to-Solution Recommender (ISR)

We design ISR; a collaborative recommendation system to recommend optimal solutions for automated network architecting. The collaborative-ness of ISR is learnt using restricted Boltzmann Machine (RBM); specifically, ISR predicts an optimal solution by inferring to the popular choices for some specific intents. ISR is trained on a crowd-source solution architecting dataset; whereby 30 the expert recommendations of solution architects are queried through a carefully designed experiment. Our intuition is that an optimal solution for a specific deployment context can be defined by the consensus of domain experts. The expert pool is made up of diverse practitioner roles to prevent localised recommendations. Besides, the selection criteria fulfil the following set of constraints: (a) participants have minimum 3 years of experience, (b) equally distributed from 4 companies {Huawei, UTAR, Jabil, Sangfor}, (c) equally pooled from 4 jobs scope {network engineer, network researchers, system admin and solution architect}. During annotations, the experts are given 50 network policies of similar policies that are slightly tweaked with different contexts and constraints. These policies

are structural different; each having different length and written in different literacy level. After evaluation, the experts then annotate one optimal solution among a set of standardise options such *access-control-list, vlan, port security, SDN, firewall, proxy, SNMP monitoring* etc. These annotation is stored as a yaml file to keep the policy ID, metadata, and the recommendations. The ‘metadata’ here refers to intent, which is resolved using SaiR from previous section. Figure 14 shows an example of the yaml file for subsequent modelling.

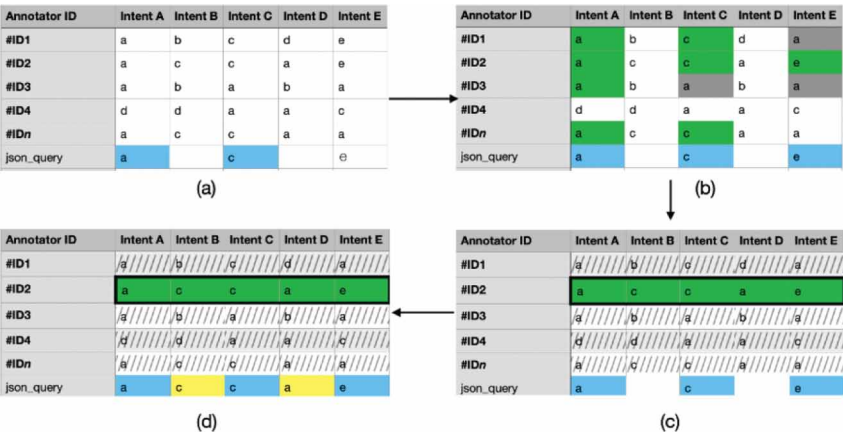
Using these crowd-source labels, we model the correlations of policy intents to suggested solution using RBM. In RBM neural network, there are no connections within visible and hidden layer. Visible and hidden layers would a fully connected layer with this restriction to top it. The RBMs calculate the probability distribution for each advertisement and send this information to hidden units that represent features. The weights learned by the RBMs are shared. We only consider annotated policy; non-annotated ones are padded with null. RBM model is capable to capture case specific constraints; whereby the algorithm find the most closely matched expert annotation to the recommend solutions for the new policies (based on higher matching intentions). Figure 15 illustrates an example of predicting optimal solutions with RBM.

ISR requires some pre-existing constraints or implementation to provide the correct ‘neighbours’ for recommendations. Consider an architecting problem to fulfil all five intents, and that the network already have two existing solutions in placed. Using collaborative filtering, ISR then finds other

Figure 14. The structure of a json query example to send prediction request to IRSR

```
policy: 28
  scope: networking
  literacy: layman
  constraints: no_cost, vendor_specific
annotator: no3, Huawei, sol_architect
id : no3
org : Huawei
role: sol_architect
intent: network_security
```

Figure 15. The RBM matrix that; the left most column denotes annotator ID, the top most row denotes the five intentions in SaiR, the rest of RBM matrix contains optimal solutions to intention mapping. In (a), a query is sent to IRSR with ‘3 pre-existing conditions’. In (b), matching neighbour are highlighted in green. In (c), the closest match neighbour is selected. In (d), IRSR recommends two solutions (shown in yellow) based on #ID2 solutions.



neighbours that have similar solutions for these two intents; and recommend appropriate solutions for the remaining three intents based on the implementational choices of these neighbours (we call this solutionID). If all existing solutions are pre-occupied, then ISR provide recommendations to replace some of these existing solutions.

4. MODEL EVALUATION

This section outline the experiment to evaluates the performance of ISR. Similar to previous annotation procedures, 20 experts from different technical backgrounds are invited to provide optimal architecting recommendation based on network policy. We design 10 synthetic case studies to cover for different implementation scenarios. ISR is used to predict for optimal solution and the results are compared with these expert recommendations. We evaluate ISR using the standard *precision*, defined below. Having a precision score, we able to estimate out of the number of the recommendations that the system provided, how many of these recommendations will be matched with respondents' preferences:

$$\text{Precision} = \frac{\# \text{ of recommended items relevant}}{\# \text{ of recommended items}} \quad (1)$$

We interpret the result in binary, such that the recommendations of our intent recogniser model (ISR) matches with the recommendation of the respondents, the prediction is true (1); if it does not matches, then it is false (0). These scores are then averaged among the 20 respondents' recommendations as the final precision for each of the case study. For example, out of 20 respondents; if 8 respondents choose A, 5 respondents choose B, 4 and 3 respondents choose C and D respectively, ISR will recommend the most popular choice. Popularity is the distribution of weighted sum of solutions for each intent. For example, if IRSR made four recommendations and two out of four of these recommendations matched the expert annotation; then, precision is calculated as:

$$\text{Precision} = 2 \text{ out of } 4 \text{ model recommendations match with respondents ranking} \quad (2)$$

$$= \frac{2}{4}$$

$$= 0.5 \text{ (the precision equivalent to 50\%)}$$

For clarity, we only show one example of the case study. IRSR expects recommendation request in the format of $\{(cond_i, cond_i+1), (int_1, int_2, int_n)\}$. The cond_i parameter states some existing solutions or legacy system that should not be replaced; meanwhile, int_i denotes the intention (from

Table 3. Calculating precision of recommendation system based on crowd source ground truth

Case ID	IRSR Recommendations	Expert Recommendations	Parity
1	A	A	1
2	A	B	0
3	C	C	1
4	B	D	0
			P=2/4=0.5

Table 4. The single test precision of IRSR in solution recommendations for one of the ten case study

Intent	IRSR Recommendations (Predicted)	Expert Recommendations (Aggregated)	Parity (1=match, 0=mismatch)
1	Adopt High Availability Mechanism	Adopt High Availability Mechanism	1
2	Device Failover	Device Failover	1
3	Avoid Single Point of Failure	Load Balancing	0
4	Perform Regular Network Maintenance	Perform Regular Network Maintenance	1
5	Load Balancing	Avoid Single Point of Failure	0
			p=3/5=0.6

SaiR) that requires a recommendation. Consider the snippet of a policy “A single network consists of multiple routers, nodes, or switches. One of those network components might become a single point of failure and stop working, which can trigger a cascade of failures within a single network.”

While the single test results indicate how fitting IRSR recommendations are; and we averaged precision score over ten case studies to measure IRSR robustness across the board. The experimental results shows that IRSR achieved 0.70 of precision score across different architectural contexts (this is measured as *coverage*). In Figure 16, the comparison of IRSR recommendations (in x) to the ground truth (in y) is visualised.

To ensure that the recommendations is not overfitting, we calculate *coverage* score to identify the ‘richness’ is the set of recommended solutions or whether the recommendations are overly repetitive. *Coverage* is calculated as:

$$\text{Coverage} = \frac{n}{N} \times 100 \quad (3)$$

N = Total recommendations in the system

n = Recommendations used

Figure 16. Comparing of IRSR recommended solutions to the ground truth (matrix); the ‘ \checkmark ’ sign denotes experts recommendation, while ‘ Ω ’ denotes IRSR recommendation

Case Study	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
ID#1	\checkmark	Ω	Ω	\checkmark	\checkmark															
ID#2							\checkmark	Ω	Ω	\checkmark	Ω							Ω		Ω
ID#3		Ω		Ω								Ω								
ID#4								Ω	Ω			Ω	Ω	\checkmark	\checkmark	\checkmark	\checkmark			
ID#5				Ω		Ω												Ω	Ω	
ID#6											Ω	Ω	Ω	Ω	\checkmark	\checkmark	Ω			
ID#7	\checkmark	Ω	Ω	Ω	Ω	Ω						Ω								
ID#8							Ω	Ω	Ω	Ω	Ω									
ID#9				Ω							Ω				Ω	Ω		\checkmark	\checkmark	
ID#10		Ω		\checkmark	Ω		Ω					\checkmark								

There are 23 recommendations in total for all the intents. IRSR successfully recommend all possible solutions (23/23); with 1.0 coverage. On each query, IRSR computes for intent, it retrieved all the recommendations consist in the list and there are no leftover recommendations. We find that despite high coverage, there could be some solutionsID that are more relatively more dominant. Intuitively, some solutionsID would be more frequently inferred to address some common generic policies as compared to context specific solutions that are found on the rather small number of niche policies.

Next, we run ‘controlled’ experiment where intents and some pre-existing conditions are selectively randomised prior to recommendations to measure IRSR’s personalisation factor. *Personalization (Ps)* is a metric access if a model recommends different solutions to different policies that have are contextually unique. The contexts here refers to constraints, cost, solution interoperability or business restrictions. Recommendation system with high Ps factor is useful for case specific use, meanwhile a low Ps factor is useful for general purpose recommendations. specifically, we build a few policies variants from the baseline policy stating ‘*social network usage during office hours are not allowed*’: (a) *social network usage during office hours are not allowed in meeting’s room* (b) *social network usage during office hours are not allowed except for lunch hours* (c) *social network (like Facebook but not WhatsApp) usage during office hours are not allowed*. We infer on IRSR to provide recommendations for each of these variants. From raw observation, we find that IRSR is does not understand fine-grain semantics. *Personalization* can be measured in a $n \times n$ matrix; like the 5x5 matrix shown in Table 5 to represent the recommended solution for our example.

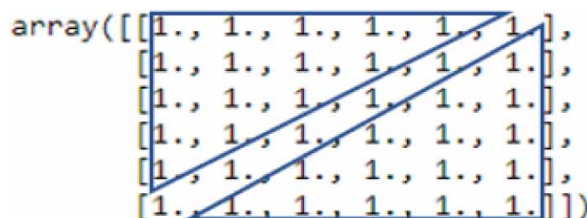
We use cosine similarity to calculate the similarity matrix (see equation 3). In Figure 17, observed that the matrix was separated by an upper triangle and a lower triangle to make a comparison between them.

We deduce that there is no difference between the value in the upper triangle and the lower triangle (denoted as blue triangle in Fig15). IRSR return the same set of recommendations despite different policies variants are queried, implying that IRSR is not accustomed to fine grain changes in policies.

Table 5. The IRSR’s recommendations on 5 policies; whereby variants a,b,c,d is derived from the baseline policy

	Proxy Web Control	Website Blocking	Access Control List (ACL)	Active Monitoring	Firewall	Keyword Blocking
Baseline	1	1	1	1	1	1
Variant (a)	1	1	1	1	1	1
Variant (b)	1	1	1	1	1	1
Variant (c)	1	1	1	1	1	1
Variant (d)	1	1	1	1	1	1

Figure 17. Measuring ‘contextual-awareness’ (personalization factor) of IRSR using similarity matrix



5. CONCLUSION

In this paper, a semantic aware recognition model is designed to understand and extract intention from networking policies for solution architecting. The proposed IRSR is a holistic pipeline that predicts the underlying intention of network policies using SaiR and correspondingly suggest appropriate solutions using ISR. SaiR use NLP algorithms to synthesise the non-overt intentions embedded in ICT policies that are often ambiguous and conflicting when written by people with different technical literacy. SaiR trains an ‘intent’ model on a constrained dataset that contains only limited keywords annotated by domain experts. SaiR use this model to extrapolate of *words to intents* correlations then automatically categorise wordings in the sentences into their respective ‘bins-of-words’ and then classifies policy into different intent classes based on the weighted sum of these bins. Meanwhile, ISR is a collaborative filtering recommender designed to provide solution recommendations based on a crowd-source ground truth. The ‘crowds’, made up of experts with many years of experience and play unique roles in solution architecting are selected to annotate the ‘most-fitting and feasible’ solutions depending on the intention context. ISR recommends from 23 possible solutions based on neighbour’s rating to cater for networking policies with both generic and niched requirements. IRSR achieved 0.7 precision and 1.0 coverage when compared to expert recommendations for various architecting scenario. The experimental results showed that IRSR successfully bridged the gap from penning policy to achieve certain business goals to actually implementing some optimal solutions to achieve these policies

ACKNOWLEDGMENT

This journal is a collaborative effort of researchers in *UTAR CIoTBD A.I. research team* working specifically in networking domain. The journal is not funded by any grants or institutions.

CONFLICTS OF INTEREST

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Process Dates:

Received: May 5, 2020, Accepted: September 16, 2020

Corresponding Author:

Correspondence should be addressed to Aun Yichiet, aunyc@utar.edu.my

REFERENCES

- Avolio, F., Fallin, S., & Pinzon, D. S. (2007). *Producing your network security policy*. WatchGuard Technologies.
- Azeem, S. A., & Sharma, S. K. (2017). Study of Converged Infrastructure & Hyper Converge Infrastructre As Future of Data Centre. *International Journal of Advanced Computer Research*, 8(5), 900–903.
- Chalon, D., Durand, Y., & Richard, B. (2001). An Overview of Automatic Network Configuration for IPv4 Appliances. In *An Overview of Automatic Network Configuration for IPv4 Appliances*, (pp. 1-16). Academic Press.
- Charalambides, M., Flegkas, P., Pavlou, G., Bandara, A. K., Lupu, E. C., Russo, A., & Rubio-Loyola, J. et al. (2005). Policy Conflict Analysis for Quality of Service Management. In *Proceedings of the Sixth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'05)*. IEEE. doi:10.1109/POLICY.2005.23
- Chaudhari, A., Asthana, A., Kaluskar, A., Gedia, D., Karani, L., Perigo, L., Gandotra, R., & Gangwar, S. (2019). Vivonet: Visually-represented, intent- based, voice-assisted networking. *International Journal of Computer Networks & Communications*, 11(2), 1–13. doi:10.5121/ijcnc.2019.11201
- Desot, T., Portet, F., & Vacher, M. (2019). Towards End-to-End spoken intent recognition in smart home. In *International Conference on Speech Technology and Human-Computer Dialogue (SpED)*. Timisoara, Romania: IEEE. doi:10.1109/SPED.2019.8906584
- Foltz, P. W. (1996). Latent Semantic Analysis for Text-Based Research. *Behavior Research Methods, Instruments, & Computers*, 28(2), 197–202. doi:10.3758/BF03204765
- Han, Y., Li, J., Hoang, D., Yoo, J.-H., & Hong, J. W. (2016). An Intent-based Network Virtualization Platform for SDN. In *12th International Conference on Network and Service Management (CNSM)*. Montreal, Canada: IEEE. doi:10.1109/CNSM.2016.7818446
- Imran, M. (2017). Open Network Operating System (ONOS) Learning Tutorial. In 29th South Asian Network Operators Group, Islamabad, Pakistan.
- Kaviani, P. (2017). Virtualization with Data Center. *International Journal of Advance Engineering and Research Development*, 4(10), 682–685.
- Latah, M., & Toker, L. (2016). Application of Artificial Intelligence to Software Defined Networking: A Survey. *Indian Journal of Science and Technology*, 9(44), 1–7. doi:10.17485/ijst/2016/v9i44/89812
- Lupu, E. C., & Sloman, M. (1999). Conflicts in Policy-Based Distributed Systems Management. *IEEE Transactions on Software Engineering*, 25(6), 852–869. doi:10.1109/32.824414
- Mallamma, V. R., & Hanumanthappa, M. (2014). Semantical and Syntactical Analysis of NLP. *International Journal of Computer Science and Information Technologies*, 5(3), 3236–3238.
- Marcon, M., Dischinger, M., & Gummadi, K. P. (2011). The Local and Global Effects of Traffic Shaping. In *Communication Systems and Networks (COMSNETS), Third International Conference on*. IEEE.
- McSweeney, A. (2019). *Introduction to Solution Architecture*. Academic Press.
- Mihaila, P., Balan, T., Curpen, R., & Sandu, F. (2017). Network Automation and Abstraction using Python Programming Methods. *6th International Conference on Recent Achievements in Mechatronics, Automation, Computer Science and Robotics (MACRo)*, 95-103.
- Nair, S. K., & Novak, D. (2007). A traffic shaping model for optimizing network operations. *European Journal of Operational Research*, 180(3), 1358–1380. doi:10.1016/j.ejor.2006.04.036
- Rajani, S., & Hanumanthappa, M. (2016). Techniques of Semantic Analysis for Natural Language Processing – A Detailed Survey. *International Journal of Advanced Research in Computer and Communication Engineering*, 5(2), 146–149.
- Saha, B., Tandur, D., Haab, L., & Podieski, L. (2018). Intent-based Networks: An Industrial Perspective. *The 1st International Workshop*.

Sameer, A. (2015). Open Network Operating System (ONOS). In *Network Operating System Seminar*. University of Babylon.

Sanvito, D., Moro, D., Gulli, M., Filippini, I., Capone, A., & Campanella, A. (2018). ONOS Intent Monitor and Reroute service: enabling plug&play routing logic. In *IEEE International Conference on Network Softwarization (NetSoft 2018) - Technical Sessions* (pp. 272-276). IEEE. doi:10.1109/NETSOFT.2018.8460064

Serdyuk, D., Wang, Y., Fuegen, C., Kumar, A., Liu, B., & Bengio, Y. (2018). Towards end-to-end spoken language understanding. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Calgary, Canada: IEEE.

Siau, K., Nah, F. F.-H., & Teng, L. (2002). Acceptable Internet use policy. *Communications of the ACM*, 45(1), 75–79. doi:10.1145/502269.502302

Singh, M. K., Rishi, O. P., Awasthi, S., Srivastava, A. P., & Wadhwa, S. (2020). Classification and Comparison of Web Recommendation Systems used in Online Business. In *International Conference on Computation, Automation and Knowledge Management (ICCAKM) Amity University* (pp. 471-480). IEEE. doi:10.1109/ICCAKM46823.2020.9051511

Wickboldt, J. A., Jesus, W. P., & Isolani, P. H. (2015). Software-Defined Networking: Management Requirements and Challenges. *IEEE Communications Magazine*, 1–8.

Yao, H., Jiang, C., & Qian, Y. (2019). *Developing Network System with Artificial Intelligence*. Springer. doi:10.1007/978-3-030-15028-0

Jasmina Khaw is an Assistant Professor at Universiti Tunku Abdul Rahman, Malaysia. She obtained her Bachelor of Computer Science, Msc and PhD (Computer Science) from Universiti Sains Malaysia (2017). She is currently working on natural language processing. Her career in academia started in 2017. Her areas of specialization include automatic speech recognition, speech synthesis and machine translation.

Ming-Lee Gan completed his Ph.D. in Computer Science studies at Universiti Tunku Abdul Rahman, Malaysia in 2013. He received his B.Eng. in Electrical & Electronics from Universiti Tenaga Nasional, Malaysia in 2004 and M.Sc. in Systems Engineering & Management from the Malaysia University of Science and Technology in 2006. In March 2012, he joined Universiti Tunku Abdul Rahman, where he is currently a lecturer. His research focus is in network path protection, network routing algorithms and network reliability analysis. A student which has a strong interest in Communication and Networking.