Individual Submission
Internet-Draft
Intended status: Experimental
Expires: 3 September 2026

Q. Li
G. Xie
Pengcheng Laboratory
Y. Jiang
Tsinghua Shenzhen International Graduate School
M. Xu
Tsinghua University

2 March 2026

In-Network Inference Protocol
draft-li-individual-inip-00

Abstract

This document specifies the In-Network Inference Protocol (INIP), a lightweight protocol designed specifically for implementing high-speed in-network inference in data center internal networks. INIP utilizes data plane devices (such as switches, DPUs, and SmartNICs) to perform lightweight inference tasks while ensuring that core network forwarding functions are not affected. The protocol operates based on the IPv4 protocol and adopts a fixed, lightweight packet format.

INIP adopts a two-tier architecture of "centralized control plane adaptation and scheduling, and minimal data plane execution". The control plane stores all inference models, deploys model rules to data plane devices using a CDN-like scheduling method, and assumes the responsibility of degraded fallback inference; the data plane performs packet parsing and match action table-based inference. This document details INIP’s core logic, packet format, data plane device constraints, model expression specifications, control plane responsibilities, CDN-like scheduling mechanism, dynamic model popularity replacement, and overall execution process.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

1.1. Background

With the rapid development of data center networks and artificial intelligence, the demand for low-latency and high-throughput inference services within data centers continues to grow. Typical scenarios include real-time traffic classification, malicious traffic detection, as well as lightweight computer vision/natural language applications (rapid product defect detection, simple face recognition, short text sentiment polarity judgment, device control command recognition, etc.).

Traditional inference deployment (deploying inference services on dedicated servers) can not use existing data plane resources for green networking and often introduces additional transmission latency. To address this, this document proposes the In-Network Inference Protocol (INIP). This protocol enables data plane devices to undertake fast inference tasks while ensuring their forwarding function. INIP makes the control plane store all models, implements in-network deployment of model rules using a CDN-like scheduling method, simplifies the adaptation of models to heterogeneous data plane devices through model distillation and cross-matching mode conversion, and optimizes data plane utilization through dynamic replacement based on model popularity. The INIP control plane also assumes the responsibility of fallback inference to ensure the normal execution of inference tasks.

1.2. Scope

This document defines INIP's core logic, packet format, data plane constraints, model expression specifications, control plane responsibilities, and overall process. The protocol is applicable to data center internal network environments, focusing on real-time, high-throughput inference scenarios that avoid large models with heavy computing. INIP does not define a new network layer protocol but operates based on the IPv4 protocol [RFC791], and model query messages initiated by users to the control plane are implemented using the lightweight DNS protocol [RFC1035] to ensure standardized adaptation.

1.3. Terminology

The key terms used in this document are defined as follows:

Data Plane Device:
A network device (such as a Tofino switch [TOFINO], DPU, or SmartNIC) that undertakes packet forwarding. A general inference table may be reserved during P4 [P4] code compilation (depending on resource availability) to execute user inference.

Inference Table
A general table reserved on a data plane device, populated with inference rules issued by the control plane. It adopts a "key field + action field" structure to perform fast match-based inference.

Feature
A core component of the key fields in the inference table, i.e., the feature data required for model inference. It corresponds one-to-one with the features in the INIP packet feature area, arranged with a fixed bit width, and serves as the core basis for the data plane to perform inference (table lookup and matching).

Matching Mode
The matching method for sample features during the operation of the data plane device’s inference table, including exact matching, range matching, and ternary matching [P4]. It is determined by the remaining hardware resources after allocation to network functions.

Control Plane
The core scheduling and adaptation node of INIP, responsible for data plane device capability management, model logic conversion, rule issuance, CDN-like scheduling, model popularity statistics, dynamic model replacement, and degraded fallback inference. It also acts as a DNS server to receive and respond to model query messages sent by users.

CDN-like Scheduling
A scheduling method that determines the deployment location of models based on topological proximity, load balancing, data plane device capability, and model popularity.

Model Popularity
An indicator measuring the frequency of model queries (only counting model queries initiated by users to the control plane via the DNS protocol), used to determine whether a model is high-frequency or low-frequency, and to provide a basis for the dynamic replacement of data plane models.

General Decision Tree

A unified model format obtained by distilling various original models (deep learning models, traditional machine learning models) [DTDISTILL].

Fallback Inference
A degradation scheme where the control plane independently executes inference tasks to ensure business continuity when there are no suitable data plane devices.

1.4. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Core Logic Overview

The core design goal of INIP is to use data plane devices to enable lightweight, fast inference in data center internal networks while ensuring that core network forwarding functions (such as routing, ACL, firewall) are not compromised. The protocol operates based on the IPv4 protocol [RFC791], with INIP packets encapsulated in IPv4 packets for transmission.

INIP adopts a two-tier architecture, with the core logic as follows:

1. Control Plane: Centrally manages all inference models (including original models and their distilled general decision trees) [DTDISTILL] [DLDISTILL], undertaking core responsibilities such as model logic conversion after distillation, inference rule issuance, device capability management, CDN-like scheduling, model popularity statistics and dynamic replacement, and fallback inference. It also receives two types of messages sent by users—model queries (implemented using the DNS protocol [RFC1035] to obtain the model's data plane deployment location) and INIP-format inference requests (to obtain fallback inference results). The control plane acts as a DNS server to parse users' DNS query requests and return the corresponding IP addresses of data plane devices that hold the specific in-network inference tasks.

2. Data Plane: Performs normal network functions and can receive model inference rules issued by the control plane. However, it only executes "packet parsing + matching-based inference" when its own IP is the destination IP of the INIP packet; after a successful inference, it needs to modify the inference result field and swap the source and destination IP pairs to ensure the packet is returned to the user.

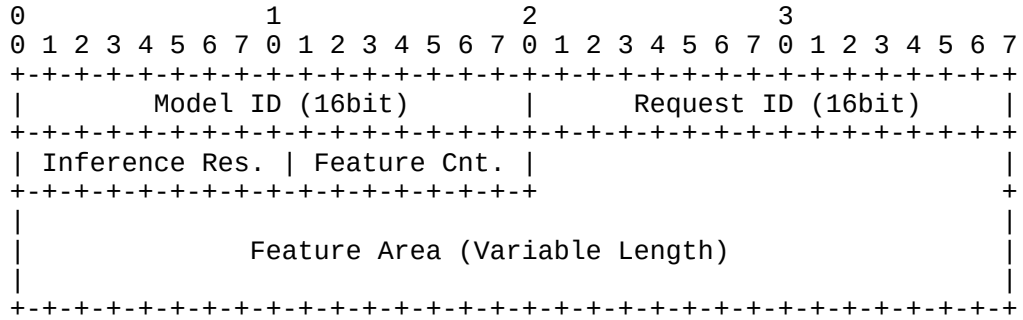
3. CDN-like Deployment and Popularity Adaptation: Based on CDN-like scheduling logic and combined with model popularity, the control plane prioritizes deploying general decision tree model rules for high-frequency queries to nearby, suitable data plane devices; the same model can be deployed on multiple data plane devices to achieve multi-node redundant deployment/load balancing, and optimize the utilization of the data plane inference table through dynamic popularity replacement.

4. Fallback Mechanism: When there are no suitable data plane devices, no corresponding model rules are deployed on the data plane, or an interruption occurs during model rule replacement, the control plane automatically executes fallback inference.

5. Request-Result Association: By setting a request identifier in the INIP packet, a one-to-one correspondence between inference requests and return results is achieved, ensuring that the user can clearly distinguish the inference results of different requests.

3. INIP Packet Format

An INIP packet consists of a fixed-length header (64 bits) and a variable-length feature area. The following is a visual representation of the INIP packet format (excluding the IPv4 header):



Detailed descriptions of each INIP packet field (all 8 bits) are as follows:

* *Model ID (16 bits)*: Uniquely identifies the inference model requested by the user, assigned by the control plane, and corresponds one-to-one with the model rules issued by the control plane to the data plane. It is the core index for users to initiate inference requests and for the data plane to perform matching-based inference; when users initiate a model query via the DNS, they need to use this model ID as the query key in the format "[Model ID].inip.local" (a local domain, resolvable only within the data center).

* *Request ID (16 bits)*: generated and carried by the user when initiating an inference request, which must be unique within a certain period. When the data plane or control plane returns the inference result, it must carry this field unchanged to ensure that the user can associate the returned inference result with the corresponding request.

* *Inference Result (8 bits)*: This field is valid when the Most Significant Bit (MSB) is set to 1; when valid, it carries the inference result returned to the user by the data plane device or control plane. When the MSB is 0, this field is a reserved bit, and the sender (user) must set it to 0, while the receiver (data plane/control plane) ignores this field. After a successful inference, the MSB of this field must be set to 1, and the corresponding inference result must be filled in.

* *Feature Count (8 bits)*: Indicates the total number of features (N) carried by the inference request, informing the data plane device of the number of features to be parsed.

* *Feature Area (Variable Length)*: Stores N feature data required for inference. Each feature adopts a globally fixed bit width (e.g., 32 bits, uniformly specified by the control plane).

4. Data Plane Device Constraints

4.1. Heterogeneous Device Support

INIP supports heterogeneous data plane devices. Regardless of the physical type (switch, DPU, smartNIC), manufacturer, or hardware specifications of the device, all data plane devices can reserve a general inference table (optional, only considered after meeting normal networking functions). Inference operations are completed through this single inference table, and hardware differences between devices do not affect protocol execution.

4.2. Data Plane Inference Table Structure

Data plane devices only reserve one general inference table, which is populated by the control plane with inference rules converted from general decision trees. The structure is simple and fixed, adapting to the data plane’s line-rate table lookup requirements. The inference table adopts a fixed "key field + action field" structure:

* *Key Fields*: Core matching fields, fixedly including two types of fields: ☒ Model ID (8 bits): Exactly the same as the Model ID field in the INIP packet, used to locate the model rule corresponding to the current inference; ☒ Inference Features (n fixed-bit-width features): Corresponding to the number of resolvable features n of the data plane, and features beyond the N features required by the model should be regarded as meaningless; the bit width of each feature is consistent with the INIP packet feature area (uniformly specified by the control plane).

* *Action Fields*: Action instructions executed after a successful table lookup, fixedly including one type of core field, namely the Inference Result (8 bits): the inference conclusion corresponding to the set of key fields (Model ID + Features), used to fill the INIP packet.

4.3. Fixed Inference Parameters

The fixed inference-related parameters of the inference table include:

* *Resolvable Feature Count (n)*: The number of fixed-bit-width features that the device can resolve. The bit width of all features is globally uniform, which needs to adapt to the value range (0-255) of the feature count (8 bits). The control plane will select suitable devices to deploy model rules based on this parameter, ensuring that n the number of features N required by the model.

* *Inference Table Matching Mode*: The matching mode (exact matching, range matching, ternary matching) of each key field in the inference table, determined by the allocation of the device’s hardware resources. For example, when the TCAM resources of the Tofino chip are prioritized for core network functions and SRAM has redundancy, the inference table will adopt an SRAM-based exact matching mode.

* *Maximum Number of Inference Table Entries*: The maximum number of inference rules that the device can carry, determined by the hardware resources of the inference table (such as SRAM capacity).

The inference-related parameters of data plane devices are globally fixed after P4 code compilation [P4]. If inference-related parameters need to be adjusted, the control plane can modify and recompile the P4 code as needed, and synchronously update the corresponding data plane device information in the capability database to ensure the accuracy of subsequent model replacement and adaptation.

4.4. Data Plane Execution Logic

Only the destination of the INIP packet (which may be a data plane device, control plane, or user) will execute "packet parsing + table-based inference (if it is a data plane/control plane) + post-hit processing" operations; network devices with non-destination IPs (including non-destination data plane devices) only perform behaviors consistent with network forwarding logic. When the destination is a data plane device, the specific execution logic is as follows:

1. Packet Parsing: Parse the INIP packet encapsulated in the IPv4 packet, extract the Model ID, Request ID, Feature Count N, and N inference features, and record the source IP address and destination IP address;

2. Matching-based Inference: Perform table lookup operations according to the inference table matching rules based on the extracted Model ID and inference features, distinguishing between hit and miss scenarios;

3. Miss Handling: If there is no hit in the inference table, forward the complete packet to the control plane (while modifying the destination IP in the IPv4 header to the control plane IP), and the control plane executes fallback inference;

4. Post-hit Processing (Core Operation): If there is a hit in the inference table, perform two key operations: ☒ Modify the "Inference Result" field of the INIP packet: Set the Most Significant Bit (MSB) of this field to 1 to identify that the inference result is valid, and fill the matched inference result from the inference table into this field; ☒ Adjust the IPv4 header: Swap the source IP address and destination IP address, i.e., change the original source IP (user IP) in the IPv4 header to the destination IP, and the original destination IP (data plane device IP) to the source IP, ensuring that the data packet can be returned to the requesting user along the original path;

5. Packet Forwarding: Send the modified IPv4 packet according to the networking forwarding logic.

5. Model Expression Specifications

All models in INIP are represented in the form of general decision trees, which are obtained by distilling various original models (including deep learning models and traditional machine learning models) through knowledge distillation technology [DTDISTILL]. This ensures the unity and adaptability of model expression, facilitating the control plane to convert them into inference rules supported by data plane devices.

A general decision tree only defines the business logic judgments required for inference (such as feature interval judgment, equality judgment) and is not bound to the matching mode of any data plane device. This allows models from different sources and types to be flexibly adapted to devices with different matching modes after distillation. The control plane stores all original models and their corresponding general decision trees, while the data plane only stores the inference table rules converted by the control plane; the control plane uniformly assigns Model IDs to ensure their uniqueness, and this Model ID will serve as the core key for users to initiate queries.

6. Core Responsibilities of the Control Plane

The control plane is the core adaptation and scheduling node of INIP, undertaking the following key responsibilities, including core functions such as CDN-like scheduling, model popularity management, and fallback inference, to ensure the efficient operation of the protocol:

6.1. Data Plane Device Capability Management

The control plane maintains a dynamically updatable device capability database, which records the fixed parameters (n, matching mode, maximum number of entries), topological location, and remaining inference table capacity of each data plane device. After a data plane device deploys or deletes model rules, or updates inference parameters, the control plane updates the corresponding device information to support CDN-like scheduling and model popularity replacement.

6.2. Model Logic Conversion

The control plane converts the business logic of the general decision tree into inference rules according to the matching mode of the target data plane device. It also assigns a unique Model ID to each model, associating the model with the inference rules. The conversion process must follow the complete logical equivalence principle and allow cross-matching mode conversion (such as converting the range logic of the decision tree into exact matching or ternary matching rules) to ensure the accuracy of the inference results; the converted rules must adapt to the "key field + action field" structure of the data plane inference table.

6.3. Rule Issuance Verification

Inference rules will be issued to the target data plane device only if both of the following conditions are met:

1. The number of resolvable features n of the device the number of features N required by the model; the control plane will only issue the corresponding model rules to devices that meet this condition;

2. The number of rules generated after conversion does not exceed the remaining number of entries in the device’s inference table.

6.4. CDN-like Scheduling and Model Popularity Management

6.4.1. CDN-like Scheduling Principles

During scheduling, the four-dimensional principles of "topological proximity + load balancing + capability adaptability + popularity priority" are followed. Priority is given to selecting devices that are close to the user, have low load, and meet the adaptation conditions (nN, number of rules remaining capacity) to deploy high-frequency popular model rules to data plane devices; the same model can be deployed on multiple suitable data plane devices to achieve near-user response and load sharing of inference requests, consistent with the core logic of CDN.

6.4.2. Model Popularity Considerations

The control plane statistics the model query messages initiated by users to it (only counting model query messages sent by users, not INIP-format inference requests sent by users), and determines the model popularity level and replacement rules in combination with data plane device constraints. The specific consideration dimensions are as follows:

* *Core Consideration Indicators*: ☒ Model Query Frequency: Count the number of queries for each model within a fixed period (e.g., 1 hour) to distinguish between high-frequency (popular) and low-frequency models (example: 1000 queries per hour are high-frequency, <1000 queries per hour are low-frequency); ☒ Request Latency Requirement: Prioritize high-frequency models that are "latency-sensitive" (such as device control command recognition, real-time traffic classification), which need to be deployed to the data plane first; ☒ Model Resource Occupation: Consider the number of data plane inference table entries occupied by model rules, and prioritize models with "high popularity + low resource occupation" to improve resource reuse rate.

* *Popularity Judgment and Dynamic Replacement Rules*: The control plane maintains a "model popularity ranking", which is updated every fixed period. The core scenario triggering dynamic replacement is "low-frequency models occupying resources while high-frequency models have no deployment space". During replacement, the principles of "high-frequency replacing low-frequency, low-resource-occupation replacing high-resource-occupation, and latency-sensitive models being retained first" are followed; if an inference request interruption occurs during replacement, the control plane automatically executes fallback inference to ensure business continuity; after the replacement is completed, inference requests are automatically switched to data plane response.

6.4.3. Model Resource Reuse

If a data plane device still has remaining capacity in the inference table after deploying one model, it can continue to deploy other models, prioritizing high-frequency popular models to achieve efficient reuse of device resources; the same model can be deployed on multiple suitable data plane devices to further improve near-user response efficiency, achieve load sharing, and reduce the fallback inference pressure on the control plane.

6.5. Degraded Fallback Inference

If there are no suitable data plane devices (e.g., no devices meet nN or the number of rules exceeds the limit), no corresponding model rules are deployed on the data plane (no corresponding Model ID), or an interruption occurs during model rule replacement, the control plane receives the INIP-format inference request message sent by the user and independently executes degraded fallback inference to ensure the inference task is not interrupted. Fallback inference has no data plane hardware acceleration, and is only used as a degradation scheme, not replacing the deployment of high-frequency models on the

data plane. When the control plane returns the inference result, it must carry the Request ID and Model ID from the packet unchanged, set the MSB of the Inference Result field to 1, fill in the inference result, and swap the source IP and destination IP to ensure the packet is returned to the user.

6.6. Responding to User Requests

The control plane acts as a DNS server to respond to model query messages sent by users via the DNS protocol [RFC1035]. The message uses "[Model ID].inip.local" (a local domain, resolvable only within the data center) as the query key, carrying the Model ID; if the model has multiple data plane deployment points (i.e., rules have been successfully issued to multiple suitable data plane devices), only the IP address of the data plane device closest to the user's topological location is returned; if the model has no data plane deployment points (no suitable data plane devices or no rules issued), the control plane's own IP address is returned. After obtaining the IP address, the user can cache the mapping relationship between the Model ID and the corresponding IP address locally, with the cache validity period uniformly configured by the control plane (set via the DNS TTL field). During the cache validity period, the user does not need to repeatedly send model query messages and can directly send corresponding messages based on the cached IP address: send INIP-format inference request messages to data plane devices, and send INIP-format inference request messages to the control plane (for fallback inference); after the cache expires, the user resends a DNS model query message to the control plane.

7. Overall Process

The overall execution process of the INIP steps is as follows:

1. The control plane compiles P4 code for each data plane device (the code first includes network forwarding behaviors and an inference table when device resources are sufficient); when there is an inference table, it updates the corresponding information in the device capability database. If the inference table parameters (number of features, table capacity) need to be adjusted later, the control plane must recompile the P4 code and update the device capability database.

2. The control plane distills and converts the original learning models into general decision trees, assigns a unique Model ID to the model, and stores all original models, corresponding general decision trees, and Model IDs; it also configures DNS services to support users to initiate queries with "[Model ID].inip.local" as the key.

3. The control plane parses the general decision tree, and screens out multiple suitable target data plane devices (the same model can be deployed on multiple suitable devices) in combination with CDN-like scheduling logic (topological proximity + load balancing + capability adaptability + popularity priority).

4. The control plane converts the decision tree logic into inference rules supported by the target device in an equivalent manner (i.e., adapting to the "key field + action field" structure of the data plane inference table), and checks whether the number of rules exceeds the limit based on the current remaining capacity of the device’s inference table; if the remaining capacity is sufficient, completes the rule issuance, and updates the remaining capacity of the device in the device capability database, as well as the deployed model rules and Model ID information.

5. The control plane statistics the frequency of user DNS model query messages corresponding to each Model ID, updates the model popularity ranking every fixed period; for high-frequency popular models, they can be deployed on multiple suitable data plane devices. If dynamic replacement of data plane model rules is needed, the replacement operation is performed. During the replacement, the control plane receives the INIP-format inference request message sent by the user and executes fallback inference.

6. The user sends a model query message to the control plane via the DNS protocol, using "[Model ID].inip.local" as the query key to obtain the corresponding IP address: if the model has multiple data plane deployment points, the control plane returns the IP address of the data plane device closest to the user; if there are no data plane deployment points, the control plane returns its own IP. After obtaining the IP address, the user caches the mapping relationship between the Model ID and the IP address locally (the cache validity period is controlled by the DNS TTL field). During the cache validity period, the user directly sends an INIP inference request message to the corresponding node based on the cached IP address; after the cache expires, the user resends a DNS model query message to the control plane.

7. Only the destination of the INIP packet (data plane device, control plane, or user) parses the packet; network devices with non-destination IPs only forward the packet normally, performing behaviors consistent with network logic.

8. If the destination is a data plane device, extract the Model ID, Request ID, and feature data, and perform table-based inference; if there is a hit in the inference table, modify the Inference Result field (set MSB to 1 + fill in the inference result), swap the source and destination IP pairs in the IPv4 header, and forward the packet back to the user; if there is no hit in the inference table, forward the original INIP packet to the control plane for degraded fallback inference (the packet destination is modified to the control plane at this time).

9. As a possible destination IP of the INIP packet, the control plane receives the INIP-format inference request message sent by the user, executes fallback inference, sets the MSB of the Classification Result field to 1, fills in the inference result, swaps the source and destination IP pairs in the IPv4 header, carries the Request ID and Model ID unchanged, and forwards the packet back to the user.

10. When the user is the destination IP, parse and receive the inference result; network devices with non-destination IPs only forward the INIP packet normally.

8. IANA Considerations

This document does not require any resource allocation from the Internet Assigned Numbers Authority (IANA). INIP reuses the private allocation value of the IPv4 Protocol field (range 128-255), which is a private use range reserved by IANA, and no IANA public allocation number is required; when users initiate model queries via the DNS protocol, the local domain "inip.local" (non-public domain) is used, which does not require domain name allocation from IANA and only needs to be configured for resolution within the data center.

9. Security Considerations

INIP is specifically designed for data center internal network environments. Users should ensure that the environment has security protection measures such as physical isolation and access control. Therefore, INIP packets are transmitted in plain text to prioritize low latency processing performance. For model query messages initiated by users via the DNS protocol, access control can be implemented through the internal DNS server of the data center to restrict only authorized users from initiating queries, preventing the leakage of Model IDs and device IP information.

10. References

10.1. Normative References

[RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <https://www.rfc-editor.org/rfc/rfc791>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/rfc/rfc1035>.

[DTDISTILL] Xie, G., Li, Q., Dong, Y., Duan, G., Jiang, Y., and J. Duan, "Mousika: Enable General In-Network Intelligence in Programmable Switches by Knowledge Distillation", INFOCOM IEEE Conference on Computer Communications, DOI 10.1109/INFOCOM48880.2022.9796856, 2022, <https://doi.org/10.1109/INFOCOM48880.2022.9796856>.

[P4] The P4 Language Consortium, "P4_16 Language Specification", Version 1.2.5, 2024.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

10.2. Informative References

[TOFINO] Intel/Barefoot Networks, "Tofino Programmable Switch Chip Datasheet", 2018.

[DLDISTILL] Hinton, G., Vinyals, O., and J. Dean, "Distilling the Knowledge in a Neural Network", NIPS Deep Learning Workshop, 2015.

Authors' Addresses

Qing Li
Pengcheng Laboratory
Email: liq@pcl.ac.cn

Guorui Xie
Pengcheng Laboratory
Email: xgr19@tsinghua.org.cn

Yong Jiang
Tsinghua Shenzhen International Graduate School
Email: jiangy@sz.tsinghua.edu.cn

Mingwei Xu
Tsinghua University
Email: xumw@tsinghua.edu.cn