

Visual Genealogy of Deep Neural Networks

Qianwen Wang¹, Jun Yuan, Shuxin Chen², Hang Su, *Member, IEEE*, Huamin Qu, and Shixia Liu

Abstract—A comprehensive and comprehensible summary of existing deep neural networks (DNNs) helps practitioners understand the behaviour and evolution of DNNs, offers insights for architecture optimization, and sheds light on the working mechanisms of DNNs. However, this summary is hard to obtain because of the complexity and diversity of DNN architectures. To address this issue, we develop DNN Genealogy, an interactive visualization tool, to offer a visual summary of representative DNNs and their evolutionary relationships. DNN Genealogy enables users to learn DNNs from multiple aspects, including architecture, performance, and evolutionary relationships. Central to this tool is a systematic analysis and visualization of 66 representative DNNs based on our analysis of 140 papers. A directed acyclic graph is used to illustrate the evolutionary relationships among these DNNs and highlight the representative DNNs. A focus + context visualization is developed to orient users during their exploration. A set of network glyphs is used in the graph to facilitate the understanding and comparing of DNNs in the context of the evolution. Case studies demonstrate that DNN Genealogy provides helpful guidance in understanding, applying, and optimizing DNNs. DNN Genealogy is extensible and will continue to be updated to reflect future advances in DNNs.

Index Terms—Interactive visual summary, information visualization, educational tool, deep neural networks

1 INTRODUCTION

OVER the past few years, deep neural networks (DNNs) have exhibited state-of-the-art performances in many applications, including classification [59], language translation [44], [65], and speech recognition [5], [24]. As a breakthrough in artificial intelligence, DNNs have been attracting attention not only from researchers, but also from practitioners who are interested in applying DNNs to their applications.

Even though DNNs have achieved impressive progress, their working mechanisms remain unclear, especially to average practitioners. Principled rules for the design of DNNs (e.g., what architecture is the best choice for a specific task) remain lacking. Researchers keep exploring new DNN architectures on the basis of their empirical knowledge, expertise, and extensive experiments, thereby leading to the wide high diversity of existing DNNs. Practitioners, especially new beginners, can be easily baffled by the variety of available DNNs and often fail to understand them and apply them as needed. Therefore, an educational tool that provides informative guidance for understanding and applying DNNs is necessary.

To inspire and motivate the wide adoption and extensive use of DNNs, a summary and presentation of existing DNNs are needed. However, summarizing DNNs and representing this summary are technically demanding. Three main challenges need to be addressed. The first challenge is

caused by *the rising number of DNNs*. Due to their excellent performances, an increasing number of DNNs have been developed for different applications [6]. It is non-trivial to examine the numerous existing DNNs and summarize representative DNNs from them. Moreover, informative visual guidance is needed to prevent users from being overwhelmed when exploring these DNNs and thus make informed decisions when choosing which DNN to use for their tasks. The second challenge comes from *the complexity of DNN architectures*. Advances in DNNs have led to complex architecture designs, including deep layers (e.g., over 1200 layers [28]), multiple branches [34], and dense skip connections [27]. Such complex architectures render the observation, comparison, and understanding of DNNs difficult, thus posing unique challenges for representation. The third challenge lies in *the diversity of DNNs*. Given the unclear working mechanisms of DNNs, DNN researchers and practitioners keep exploring various DNNs for different tasks, leading to a variety of DNN architectures. The diversity of DNN architectures results in challenges in identifying the evolutionary relationships among DNNs (e.g., one DNN is inspired by another DNN) and in revealing the evolution patterns of DNNs (e.g., DNN architectures are getting deeper).

To tackle the aforementioned challenges, we design and implement DNN Genealogy, a visual educational tool, to facilitate the exploratory analysis of different DNNs and to understand the pros and cons of each DNN in the context of model architecture and performance. To summarize the large number of existing DNNs, we employ a semi-automatic survey approach to extracting representative papers from a total of 16465 papers, constituting all papers published in CVPR, ICLR, NIPS, IJCAI, ICML, AAAI, and JMLR from 2012 to 2018. A hierarchical topic model [10] is employed to aid in the analysis of such massive literature. A four-step pipeline is developed to extract 140 relevant papers, from which we

- Q. Wang and H. Qu are with the Hong Kong University of Science and Technology, Kowloon, Hong Kong. E-mail: qwangbb@connect.ust.hk, huamin@cse.ust.hk.
- J. Yuan, S. Chen, H. Su, and S. Liu are with the Tsinghua University, Beijing 100091, China. E-mail: {yuan-j15, csx15}@mails.tsinghua.edu.cn, {suhangss, shixia}@tsinghua.edu.cn.

Manuscript received 2 Jan. 2019; revised 30 Apr. 2019; accepted 31 May 2019.
Date of publication 6 June 2019; date of current version 6 Oct. 2020.
(Corresponding author: Shixia Liu.)
Recommended for acceptance by E. Eismann.
Digital Object Identifier no. 10.1109/TVCG.2019.2921323

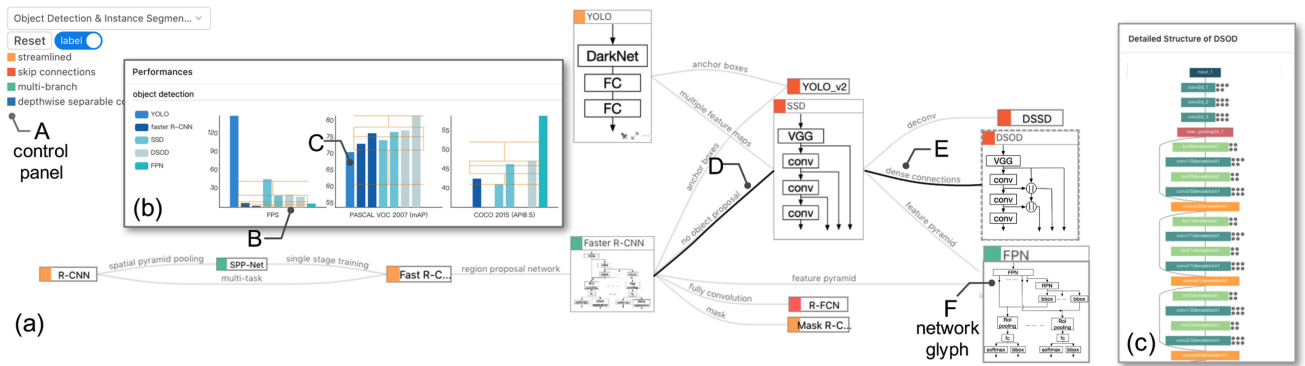


Fig. 1. DNN Genealogy is an interactive visualization tool that provides a visual summary of representative DNNs. (a) the *Evolution View* illustrates representative DNNs and the evolutionary relationships among them; (b) the *Performance View* compares the performances of different DNNs. (c) the *Architecture View* conveys the detailed architecture of a DNN.

manually identify 66 representative DNNs. Consulting other surveys [21], [35], [67] and discussing with DNN experts further validate our summary. To assist in understanding and learning DNNs, we design and implement DNN Genealogy to present our summary and analysis of these 66 DNNs, as shown in Fig. 1. The evolutionary relationships among the DNNs are illustrated by a directed acyclic graph (DAG), where textual annotations provide detailed explanations of these relationships. A focus+context visualization based on a degree of interest (DOI) algorithm is developed to facilitate the exploration of these DNNs by directing users to relevant content. To help users quickly learn the main characteristic of a complex DNN architecture, we design a set of network glyphs to represent the collected DNNs based on the original papers and our understanding.

Moreover, on the basis of our development and the case studies of DNN Genealogy, we derive the following two findings. First, we find that most DNNs can be categorized into nine types according to their architectures. Second, we identify a set of patterns in the evolution of DNNs, including: 1) DNN architectures are getting deeper and wider; 2) skip connections are widely adopted and are getting denser; 3) new architecture types are quickly combined with existing architecture types by follow-up work to improve the performance further; 4) many DNNs are proposed by combining the advantages of different existing architectures. We hope these findings can promote the understanding of DNNs and offer insights that help users choose, modify, and design their own DNNs.

The scope of this work is focused on supervised learning and discriminative models, given their dominance over recent real-world applications. A demo of DNN Genealogy is available at <http://dnn.hkustvis.org/>. The main contributions of this work are as follows:

- A summary and analysis of 66 representative DNNs from a systematic survey of existing DNNs;
- A summary of the design requirements for developing DNN educational tools based on interviews and questionnaires with DNN practitioners;
- The design and implementation of DNN Genealogy, an educational tool that provides informative visual guidance in understanding and applying DNNs.

2 RELATED WORK

In this section, we briefly review two categories of work related to ours: summarization of DNNs and visualization of DNNs.

2.1 Summarization of Deep Neural Networks

A number of manual survey efforts have been made to summarize existing DNNs, providing guidance for learning the basic DNN architectures under different categories, typical work in each category, and their applications in different fields.

Most survey papers reviewed DNNs for specific applications, such as sentiment analysis [68], image segmentation [20], and action recognition [6]. These papers usually require a certain level of prior knowledge about these applications and are less suitable for novices and inexperienced DNN practitioners. Moreover, these works focus on specific applications only, thereby failing to offer a comprehensive overview of the field of DNNs. To provide a comprehensible overview, Lecun et al. [35] summarized the essential concepts in deep learning, including supervised learning, back propagation, convolution networks, and distributed representation. Salehinejad et al. [53] described the development history of recurrent neural networks (RNNs) and highlighted the major advances. Although these surveys help users better understand the development of RNNs and major concepts in deep learning, they do not explicitly illustrate the evolution of DNNs (e.g., how ResNeXt [66] was inspired by ResNet [23] and GoogleNet [59]). The evolution of DNNs is useful for practitioners in understanding the connections among DNNs and would enable them to select the appropriate DNNs for their tasks.

Compared with the aforementioned work, DNN Genealogy aims to help users build a comprehensive understanding of existing DNNs, with an emphasis on their evolutionary relationships and the patterns in their evolution. Moreover, DNN Genealogy reveals DNNs and their evolution in a visual way, which has been proven to be more efficient for knowledge representation [9]. The developed visualization provides a clear overview of the DNN evolution, guides users during exploration, and supports a detailed examination of the DNNs of interest.

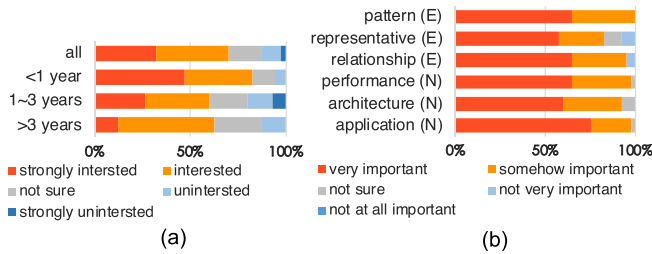


Fig. 2. Results of the questionnaires: (a) The importance rating of two types of information; (b) the distribution of interest in the developed tool. E stands for information related to the evolution of DNNs. N represents information related to one DNN.

2.2 DNN Visualization

Relevant work on DNN visualizations aims to illustrate DNNs from three aspects: architecture, performance, and hidden layer.

Architecture Visualization. The complexity of DNN architectures, including the large number of hidden layers and the variety of connections among them, leads to difficulties in understanding and communicating the architecture design. Deep learning toolkits, including Keras [14] and MXNet [11], provide tools for visualizing neural network architectures. State-of-the-art DNNs may consist of several hundreds of layers [28] and millions of connections [69], which result in large graphs that are hard to examine, understand, and compare. To demonstrate the main ideas of the proposed DNN architectures and compare them with previous work, DNN researchers usually draw diagram thumbnails for DNN architectures according to their understanding. These diagram thumbnails, instead of showing the complete architecture of a DNN, demonstrate only the modified part compared with existing architectures, thus facilitating the understanding and communication of a newly developed DNN architecture. However, these diagram thumbnails created by different authors have different styles and thus not easily compared with each other. To solve the aforementioned problems, DNN Genealogy offers a two-level visualization of DNNs: a set of network glyphs with a unified style that conveys the main characteristic of a DNN architecture and a detailed structure view that provides the demanded details.

Performance Visualization. Many visualization tools have been developed to offer detailed information about the performance of a DNN [1], [4], [64]. For example, TensorBoard [64] provides modules for monitoring scalar values (e.g., loss, accuracy) and the distribution of tensors to help users analyze and refine their networks. Moreover, performance visualization can facilitate better understanding of the working mechanisms of DNNs. Bilal et al. [8] presented Block to reveal and analyze the confusion patterns between classes. These patterns further dictate the hierarchical learning behavior of CNNs. However, these visualizations are developed to examine and analyze the performance of a single DNN and therefore cannot be directly applied to investigate and compare the performance of a large number of DNNs.

Visual Analysis of Hidden Layers. Various visual analysis techniques and systems have been developed to investigate the hidden layers of different types of DNNs and thus reveal the inner workings of these networks, including MLP [49], CNNs [29], [38], [40], [48], [49], deep generative models [39], and RNNs [47], [57]. For example, Liu et al. [40] developed

CNNVis to disclose the multiple facets of hidden layers and the interactions between them. Liu et al. [39] extracted, sampled, and visualized a large amount of time series data that represent training dynamics (e.g., activations, gradients).

Unlike the aforementioned works, DNN Genealogy facilitates the understanding of commonly used DNNs in context. It provides an overview of 66 carefully selected DNNs, the evolutionary relationships among them, their performance across typical measures, as well as their detailed architectures.

3 DESIGNING DNN GENEALOGY

To make DNN Genealogy an effective educational tool, we need to answer two questions: *What types of information are needed for learning DNNs? How should they be represented to facilitate the learning process?* Therefore, we conducted interviews and administered questionnaires to investigate current practices and the information required for understanding and applying DNNs. Grounded on the findings from the interviews and questionnaires, we derived six design requirements to guide the design and implementation of DNN Genealogy.

3.1 DNN Expert Interview

To understand current practices and identify potential opportunities for DNN Genealogy, we interviewed seven DNN experts, who are denoted by $E_i (i = 1, \dots, 7)$. These experts are senior DNN researchers from different research areas, including face detection, video analysis, natural language processing, and 3D modeling. The interviews were semi-structured and focused on two aspects: the information needed for understanding/developing DNNs and the common practice in obtaining such information. The interview lasted approximately 35 minutes for each expert.

Based on the findings from these interviews, we created a questionnaire for a larger range of survey.

3.2 Questionnaire on Understanding and Developing DNNs

We conducted a questionnaire to investigate the current practices of DNN practitioners and researchers. This questionnaire, grounded on our interviews with the aforementioned experts, was distributed to DNN practitioners within three universities and one technology company. In the survey, we targeted junior DNN practitioners (e.g., first-year PhD students, interns), who are the targeted users of DNN Genealogy. Overall, we received 40 responses. Of all the participants, 20 percent had more than three years of experience in using DNNs, 37.5 percent had one to three years, and 42.5 percent had less than one year.

According to the participants, the most common source of learning DNN architecture was textual content, including papers, online tutorials, and textbooks. Over 70 percent of all participants showed interest in an interactive visualization tool that could demonstrate the evolution of DNNs and guide them in exploring DNNs, as shown in Fig. 2a. It was not surprising that their interest decreased as their experiences in DNNs increased.

The majority of the participants reported that the understanding of existing DNN architectures was essential for them in building a DNN. Participants were asked to rate their agreement with related statements on a five-point

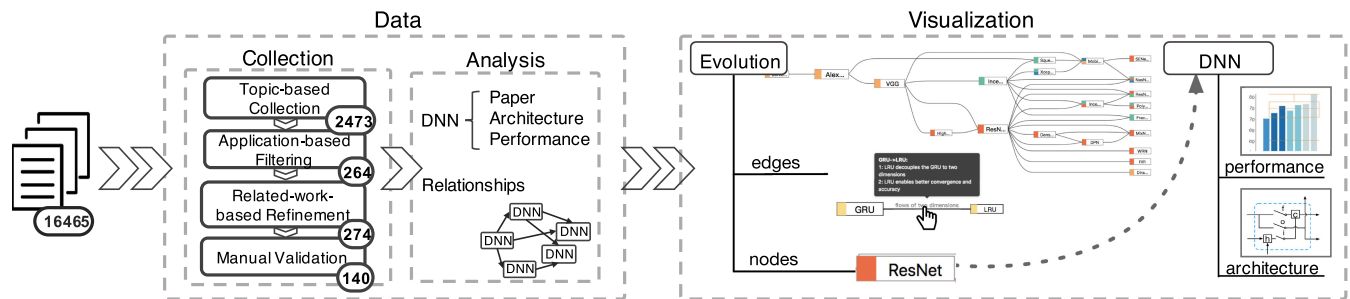


Fig. 3. DNN Genealogy consists of two main modules: the data module and the visualization module. The visualization module is composed of an evolution visualization and a DNN visualization.

Likert scale from strongly disagree to strongly agree. All participants agreed (55 percent strongly agreed, 45 percent agreed) that they surveyed relevant DNNs before developing their own DNNs. Fig. 2b shows the importance rating of different information about DNNs. The information is categorized into two classes. The first class contains information related to understanding the evolution of DNNs, including representative DNNs, patterns of evolution, and the relationships among the DNNs. The second class contains information related to understanding one DNN, including the architecture and performance of a DNN.

3.3 Design Requirements

Based on our questionnaires and interviews with DNN practitioners, we derived six design requirements, which are categorized into two classes: learning the evolution of DNN architectures (R1, R2, R3) and investigating one particular DNN (R4, R5, R6).

R1: Explaining the Relationships Among DNNs. The development of a new DNN is usually inspired by previous work. Showing how a DNN is related to previous ones helps users identify beneficial modifications and understand the advantages of different DNN architectures. More than 70 percent of the questionnaire participants reported that the relationships among DNNs were (very) important to them.

R2: Identifying the Evolution Pattern of DNN Architectures. The development and optimization of DNNs constitute a trial-and-error process. Fortunately, the abundance of DNNs offers an opportunity to identify the evolution pattern of DNN architectures (e.g., DNN architectures are becoming deeper and wider), thus offering informative guidance for applying and optimizing DNNs. As the questionnaire results suggested, all the participants (strongly) agreed that they were interested in learning the major evolution pattern of DNN architectures. Experts E4, E5, and E6 stated that they systematically analyzed existing DNNs to distill design patterns, which helped them develop their own DNNs.

R3: Identifying Representative DNNs. The increasing interest in DNNs has produced numerous DNNs. Identifying the representative DNNs from the large number of existing DNNs is needed for learning DNNs. According to the questionnaire, the majority (82.5 percent) of the participants reported that knowing representative DNNs was crucial for their learning of DNNs. Such a statement was also confirmed in the interviews with the experts. E2, E3, and E4 spontaneously mentioned the representative DNNs in their research areas during the interview and stated that understanding these

representative DNNs was essential for their understanding of the research field of DNNs.

R4: Understanding a DNN from Different Aspects. To obtain a comprehensive understanding of a DNN, analysis of different aspects of this DNN is required. The questionnaire results identified three important aspects of a DNN: architecture (regarded as important by 97.5 percent of the participants), performance (by 97.5 percent), and application (by 92.5 percent).

R5: Illustrating DNN Architectures. Understanding DNN architectures can be challenging due to their increasing complexity. Recent research has proposed DNNs that are very deep (e.g., over 1200 layers [28]) or with complex connections (e.g., combination of 13 operations [69]). Directly drawing the complete architecture of a complex DNN might result in a large graph that is difficult to examine and understand. It is, therefore, preferred to present a DNN architecture from multiple levels of details, including an abstract representation that demonstrates the main idea of this architecture (e.g., adding of skip connections) and a detailed representation of the complete architecture that provides details of each layer (e.g., the number of filters in a convolutional layer).



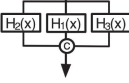

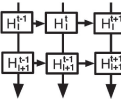
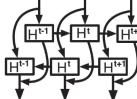
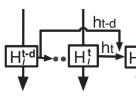
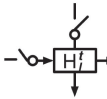
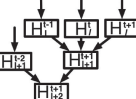
R6: Comparing Different DNNs. According to our questionnaire, 95 percent of the participants (strongly) agreed that they wanted to compare different DNNs to improve their understanding. They compared a DNN with similar DNNs to understand its architecture and the benefits brought about by its architecture. When choosing a DNN for a task, they compared different DNNs to choose the one that best met their requirements.

3.4 System Overview

Based on the collected requirements, we developed DNN Genealogy, which consists of two main modules: a data module and a visualization module (Fig. 3).

The Data Module semi-automatically extracts representative papers from the existing literature. 66 DNNs and their evolutionary relationships were finally retrieved and then fed to the visualization module for further analysis. *The visualization module* consists of evolution visualization and DNN visualization. The evolution visualization discloses the evolution of DNN architectures, which is displayed as a DAG because of its intuitiveness (R1). In the DAG, edges demonstrate the evolutionary relationships among DNNs (R2) whereas nodes denote representative DNNs (R3). The DNN visualization reveals the detailed information about a DNN architecture (R4, R5) and facilitates the comparison between different DNNs (R6).

TABLE 1
Types of DNN Architectures

CNN				RNN				
Streamline	Depthwise separable conv	Multi-branch	Skip connections	Stacked	Bidirectional	Multiple time-scales	Gated	Tree-structured
								
Layers are stacked on top of one other	A standard convolution is split into depthwise convolution and a 1x1 convolution	The output of one layer goes through multiple branches and then converges	A connection skips one or more layers	Layers are stacked to increase the depth of a RNN	A standard recurrent unit is split into two parts to process the input sequence in two directions	Recurrent units operate at multiple time scales	Add the gate mechanism	The connection graph is structured as a tree
E.g., VGG [55]	E.g., Xception [14]	E.g., Inception [58]	E.g., ResNet [22]	E.g., EESEN [45]	E.g., BRNN [53]	E.g., Clockwork RNN [31]	E.g., LSTM [24]	E.g., Tree-LSTM [59]

4 COLLECTING AND ANALYZING DNNs

A systematic collection and analysis of representative DNNs are at the core of DNN Genealogy. In this section, we describe our method of collecting and analyzing DNNs.

4.1 Collecting Papers

Inspired by the survey approaches proposed by Sacha et al. [52] and Liu et al. [41], we retrieved representative papers in a semi-automatic manner from a total of 16465 papers, including all papers published in AAAI (3274), ICML (1579), ICLR (1109), NIPS (2790), IJCAI (2571), CVPR (4255), and JMLR (887) from 2012 to 2018, to obtain a comprehensive summary of DNNs. The collection method includes the following four steps.

Topic-based Collection. We built a topic model to automatically cluster these papers based on their topics and identified a subset of papers that are related to the scope of this work. Specifically, we used hierarchical latent tree analysis (HLTA) [10] to build a hierarchical topic model from the titles and abstracts of these 16,465 papers. HLTA is employed because of its ability to discover more coherent topics and better topic hierarchies compared with the traditional latent Dirichlet analysis-based methods [10], [42]. Each topic is characterized by words that occur with high probabilities in this topic and low probabilities outside this topic. Each document belongs to different topics with varied probabilities. HLTA built a three-level topic tree, with 25, 141, 935 topics at each level. In this topic tree, we first automatically excluded irrelevant topics (e.g., bandit-problem) in a keyword-based manner. We then manually identified relevant topics to this work, including one level-1 topic (i.e., deep learning), eight level-2 topics (e.g., object detection), and thirty-five level-3 topics (e.g., layers, detector). We then extracted 2,473 papers belonging to these topics with a probability higher than 0.98.

Application-based Filtering. We filtered the collection based on three benchmark applications, that is, classification, detection, and segmentation. It is impractical to include DNNs for all applications in DNN Genealogy, especially considering that many applications are specific and have little educational value. At the same time, recent DNNs are usually highly

modularized [26], [67]. The DNN for a high-level application usually consists of several modules borrowed from benchmark applications. For example, image caption generators use CNNs borrowed from image classification to extract visual features and RNNs borrowed from language translation to generate captions [12], [63]. We identified topics explicitly related to the three benchmark applications and then extracted a list of keywords from the papers belonging to these topics. This list of keywords was used to filter the collection of 2473 papers. After the application-based filtering, we obtained 264 papers.

Related-Work-based Refinement. To avoid missing important DNNs, we adopted a rule-based method to extract papers that were mentioned as related work in the 264 papers. We identified 25 papers that were mentioned as related work but not included in the 264 papers. Among the 25 papers, 15 papers about non-DNN-based approaches were removed. The remaining 10 papers were added to our collection.

Manual Validation. To ensure that the collected papers were within the scope of our work, we manually checked these 274 papers using the following criteria. First, we removed theory papers, evaluation papers, and papers about non-DNN approaches. Second, we checked whether the contributions of the papers included DNN architectures. We also consulted other surveys [21], [35], [67] and discussed with the DNN experts to validate our collection. After the manual validation, we obtained 140 relevant papers. Note that we did not specify the data types. In addition to DNNs for euclidean data (e.g., image, text), we also included DNNs for non-Euclidean data (e.g., social network).

4.2 Analyzing DNN Architectures

Manual Paper Coding. For the relevant papers, we conducted manual coding to extract the corresponding information of the DNNs. In particular, we extracted a brief description of its architecture, motivation, and addressed problem. In addition, we iteratively identified and refined a set of architecture types. After that, we extracted nine types of commonly used architectures and 66 representative DNNs.

Architecture Types. The nine types of architectures, four of which are commonly used in CNNs and five are widely used

in RNNs, are summarized in Table 1. We formulated DNN architectures as layers and connections among these layers. One layer was represented as $H_l(x)$. We unfolded recurrent layers and used H_l^t to represent the l th layer at time t .

Evolutionary Relationships. To visually reveal the evolutionary relationships among the DNNs, we specified how one DNN is related to and inspired by previous work. We reviewed all the collected DNNs and connected two DNNs if their relationship is discussed in a paper, especially in the related work section. Explanations of these connections were extracted from the papers. For example, Chollet [15] discussed how Inception inspired him to propose Xception. Thus, we directed Inception to Xception and wrote an annotation to explain this connection. The collected DNNs were further validated by the DNN experts we interviewed, one of whom is the coauthor of this paper. DNN Genealogy has been publicly released to call for further validation.

To help users better understand the representative DNNs, we also collected related information for each DNN. One DNN is introduced based on its *architecture*, *performance statistics*, *relevant research papers*, and a *brief textual description*.

5 VISUALIZATION

In this section, we describe the two visualization components: evolution visualization and DNN visualization. Among all the visualizations, the textual annotations in evolution visualization are manually labeled, the network glyph in DNN visualization is handcrafted, and others are traditional data visualizations.

5.1 Evolution Visualization

Various graphical representations have been proposed for visualizing genealogical data [7], [30], [45]. Based on the existing literature, the design requirements of DNN Genealogy, and the characteristics of our data, we developed the evolution visualization of DNN Genealogy.

The design requirements of DNN Genealogy include explaining relationships (R1) and identifying evolution patterns (R2). Previous studies have shown that a directed acyclic graphs (DAG) could effectively show the relationships in genealogical data [30], [45]. As a result, we represent the evolutionary relationships among DNNs as a DAG using the Sugiyama-style layered graph drawing [19], [58]. In the DAG, nodes represent DNNs and edges represent the relationships among the DNNs. Textual annotations are manually labeled on the edges to explain the relationships among DNNs. We offer two types of annotations: a brief annotation that is labeled on the edges and a detailed annotation that shows when users hover over the corresponding edge.

To facilitate the exploration of DNN evolution over time, we apply a focus+context visualization. A DNN can be represented as a node or a network glyph based on its degree of interest (DOI) value. A DNN with a high DOI value is represented as a network glyph to provide an overview of the model architecture. By contrast, one with a low DOI value is represented as a node for space efficiency.

To calculate the DOI value of each DNN, we implement a DOI function based on the one proposed by Furnas et al. [18]. The DOI value of a DNN x is jointly decided by the importance

of this DNN x and the user's current interest y (i.e., the one clicked by the user), which is formulated as below.

$$DOI(x|y) = IM(x) - Dist(x, y),$$

where IM stands for the importance score of a DNN and is normalized to $[0,1]$. $Dist(x, y)$ encodes the distance between x and y . We calculate $Dist(x, y)$ as the minimal graph distance between x and y . An edge interest $EI(x, y) \in (0, 1]$ is assigned to each edge $e(x, y)$. The weight of the edge $e(x, y)$ is $1 - EI(x, y)$ when calculating the minimal graph distance $Dist(x, y)$. A large $EI(x, y)$ value means that the user is interested in x when focusing on y . We calculate $EI(x, y)$ based on the similarity between x and y , which is defined as the Jaccard similarity coefficient of the architecture types (i.e., Table 1) of the two DNNs:

$$EI(x, y) = \frac{types(x) \cap types(y)}{types(x) \cup types(y)},$$

where $types(x)$ refers to the architecture types of DNN x . For example, $types(ResNeXt)=[Multi\text{-}branch, Skip\ connections]$ and $types(ResNet)=[Skip\ connections]$. Thus, the similarity between ResNeXt and ResNet is $1/2$. The edge interest is encoded using the opacity of the edge.

We calculate the importance score of a DNN as a linear combination of the citation number per month and the average weekly interest of this DNN obtained from Google Trends [2]. Such an importance score function, even though it reveals the importance rate of a DNN, can lead to a DOI distribution with peaks, thereby making the small differences among low DOI values imperceptible. To tackle this issue, we adopt the diffusing function purposed by Van et al. [61].

$$IM_{diff}(x) = \max(IM(x), \beta \cdot \max_{n \in N(x)} (EI(x, n) \cdot IM_{diff}(n))),$$

Here, the importance score of a node depends on its maximum importance score and a fraction of its neighbor with the highest importance score. In a sense, we are diffusing the importance scores over the entire graph, where we set $\beta = 0.5$. As a result, the DOI function estimates users' potential interest in a DNN based on its importance and its relationship to the current focus DNN, thereby providing navigation cues to users during their exploration.

5.2 DNN Visualization

In this section, we discuss the visualization designs of performance statistics and model architectures, which are two important factors in learning and understanding a DNN.

5.2.1 Performance Statistics

The performance visualization was designed based on the current practice of DNN practitioners. During the interviews, we found that DNN practitioners usually compare many DNNs across several datasets. For example, Huang et al. [27] compared DenseNet with eleven existing DNNs across five datasets. We also found that DNN practitioners paid unequal attention to different DNNs when comparing them. Some DNNs were mentioned mainly to offer an awareness of the context (e.g., the average performance on a benchmark dataset). Taking the example of DenseNet [27], among the eleven DNNs discussed in [27], practitioners focused on comparing

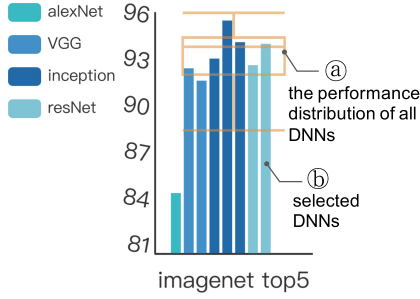


Fig. 4. A visualization that combines box plots (a) with bar charts (b) helps users compare the performance of DNNs.

DenseNet with FractalNet (the previous state-of-the-art DNN) or with ResNet (a DNN that inspired DenseNet).

Based on this observation, we combined bar charts with box plots to facilitate the process of comparing DNN performances (Fig. 4). The bars represent the performance of the DNNs that the users are interested in (i.e., the DNNs with high DOI values). The box plots represent the performance distribution of all DNNs, thereby providing an awareness of the context. The performance visualization is linked with the evolution visualization, thereby enabling users to connect the performances with the corresponding DNNs displayed on the evolution visualization, as well as to effectively locate the best-performing DNNs under different metrics.

5.2.2 DNN Architectures

Considering the complexity of DNN architectures, DNN researchers usually use diagram thumbnails to describe the main characteristic of a DNN architecture. Following this practice, DNN Genealogy enables users to understand DNN architectures from an abstract level using a handcrafted network glyph and a concrete level using a complete architecture graph. The handcrafted network glyph demonstrates the main characteristic of a DNN architecture, whereas the complete architecture graph enables users to explore details on demand.

Network Glyph. To offer users an abstract representation of different DNNs, a straightforward solution is to copy the diagram thumbnails from the original papers. However, these diagrams thumbnails, which were drawn by different authors in various styles, can impede comparison and understanding when users explore the evolution of DNNs over time (Fig. 5). Thus, we borrow lessons about network glyph designs from existing DNN papers and develop a set of network glyphs with a unified style to facilitate the

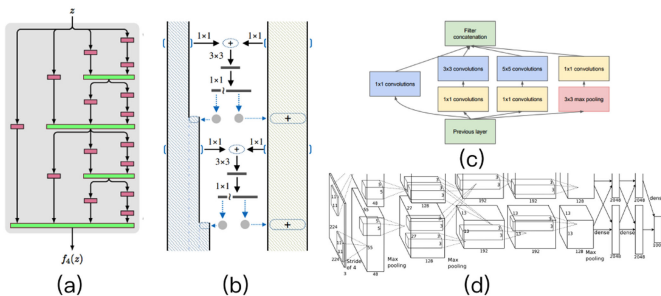


Fig. 5. Diagram thumbnails of different styles: Four example diagram thumbnails from FractalNet (a) [33], Dual Path Networks (b) [12], GoogleNet (c) [58], and AlexNet (d) [32].

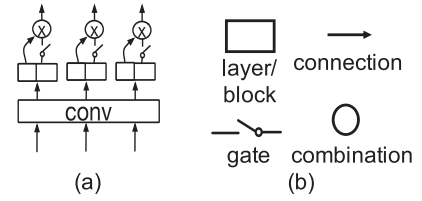


Fig. 6. The proposed network glyphs: (a) The network glyph for linear gated units [15]; (b) graphical representations of different components.

understanding and comparison of different architectures. Users can compare the glyph of two adjacent DNNs by hovering on the edge that connects the two DNNs.

The design of network glyph follows three principles. The first one is concise. The network glyph aims to provide an abstract presentation of a DNN architecture and help users quickly understand the main characteristic of a DNN architecture. The second one is intuitive. The network glyph should be intuitive and can be easily understood by the users of DNN Genealogy. The third one is generality. The network glyph should be able to be generalized to commonly used DNN architectures to make DNN Genealogy extensible. Based on these three principles, we identify four key components (i.e., layers/blocks, connections, gates, and combinations) in commonly used DNN architectures and represent them graphically, as shown in Fig. 6. The network glyph is developed based on the assembly of the graphical representations of these four key components. We modify the diagram thumbnails in the collected papers if they were available and matched the aforementioned formulation. Otherwise, we handcraft the network glyphs ourselves.

Architecture Visualizer. We follow previous practice [14], [64] and visualize the complete architecture of a DNN as a DAG. Our implementation uses Sugiyama-style layered graph drawing [58], which extends the architecture visualizer built in Keras [14], a popular and user-friendly deep learning framework. First, we add interactions to enable users to examine the details (e.g. number of channels) of each layer. Second, we illustrate the number of parameters at each layer using a dot-based visualization, where the number of dots represents the logarithmic value of the number of parameters. Third, we add the comparison of DNN architectures.

5.3 Design Iterations

To investigate the usability of our design, we performed usability sessions three times with four postgraduate students (three reported themselves as novices and one as an intermediate in DNNs) during the development of DNN Genealogy. Our modifications in the iterative design process can be summarized into two categories: encouraging exploration and facilitating understanding.

Encouraging Exploration. We implemented a DOI-based focus+context visualization to encourage exploration in DNN Genealogy. Initially, we showed all DNN as nodes and the network glyph showed only when users clicked a DNN. However, some users quickly stopped the exploration and expressed their needs for guidance in further exploration. Therefore, we implemented a DOI function to calculate the DOI values of DNNs in accordance with the users' current interest. DNNs with high DOI values were then shown as

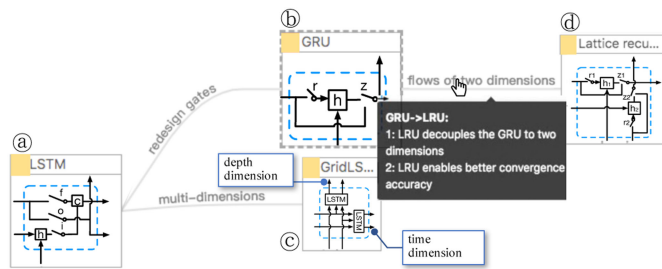


Fig. 7. Comparing the network glyph of LSTM (a), GRU (b), GridLSTM (c), and Lattice Recurrent Units (d). GRU redesigns the gates in LSTM. Lattice Recurrent Units extend GRU, and GridLSTM extends LSTM by dividing the one-dimensional data flow into two dimensions: time dimension (i.e., horizontal direction) and depth dimension (i.e., vertical direction).

network glyphs as a visual cue to direct attention and to encourage further exploration.

Facilitating Understanding. To facilitate understanding, we reduced the use of color encoding and provided additional textual explanations. Initially, we used color intensively to encode different types of information. However, users reported that the extensive use of color was confusing. For example, some users confused the color used to encode layers in the network glyphs with the one used to encode architecture types. As a result, we removed the color encoding in the network glyphs. Meanwhile, based on users' feedback, we provided detailed textual explanations in the *Info View* and in the hover-on tooltips on the edges of the *Evolution View*.

6 CASE STUDY

6.1 Understanding DNNs

This case study demonstrated that DNN Genealogy could help users, especially those with limited knowledge of DNNs, understand DNNs. We collaborated with two undergraduate students (S1 and S2) majoring in computer science. They had basic knowledge about DNNs and were interested in acquiring additional knowledge through DNN Genealogy.

6.1.1 Exploring and Learning DNNs

S1 was interested in DNNs for sequence modeling and decided to use DNN Genealogy to learn these DNNs.

Learning LSTM (R4). S1 had heard about LSTM before but had unclear information about its architecture and working mechanism. Hence, she started her exploration from LSTM. From the network glyph of LSTM (Fig. 7a), S1 got a basic idea about the architecture of LSTM, which consists of different gates. With the description offered in the *Info View*, S1 learned the functionality of these gates, namely, to accumulate previous information and to avoid gradient vanishing/exploding. S1 further explored the related links offered in *Info View* and showed special interest in *LSTM: a search space odyssey*, which studies the computational components of LSTM variants. S1 saved this paper and said that she would read it later.

From LSTM to GRU: Understanding their Relationships (R1). S1 then switched back to the *Evolution View* to continue her exploration. Next to LSTM, the network glyph of GRU was also expanded (Fig. 7b). DNN Genealogy estimates that the user was interested in GRU based on her current focus on LSTM. By comparing the network glyph of LSTM with that of GRU, which are displayed in a unified style and placed side by side, S1 identified the differences between their

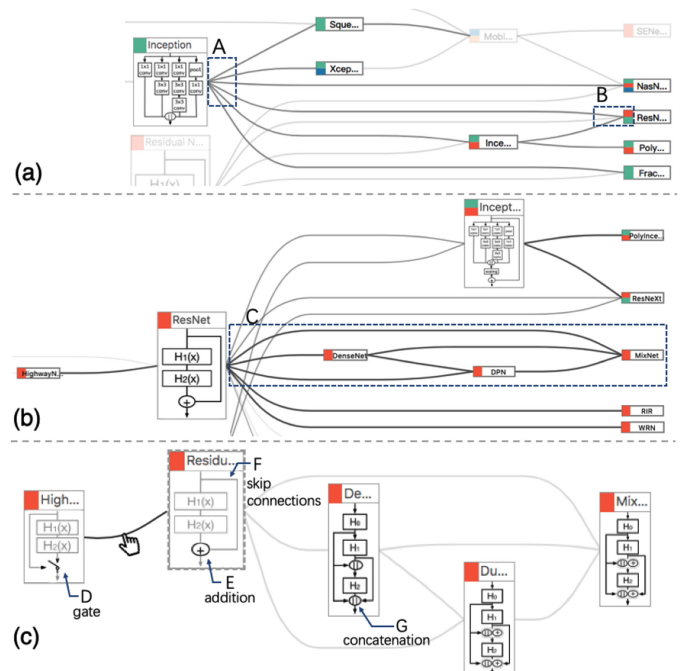


Fig. 8. DNN Genealogy illustrates the evolution of different types of DNN architectures: (a) the evolution of *multi-branches*; (b) the evolution of *skip connections*. The network glyphs enable users to compare different DNN architectures in the context (c).

architectures: “GRU has two gates while LSTM has three gates” and “GRU only has one input from the previous state while LSTM gets two.” Her opinion was confirmed by the label “*re-design gates*” on the edge connecting LSTM with GRU. Hovering on the edge revealed a detailed explanation of the evolutionary relationship between LSTM and GRU, which justified the motivation of GRU (i.e., “*improve computation efficiency*”) and its method (i.e., “*combining forget gate and input gate into one single gate*”).

Examining DNN Relationships to Learn Beneficial Modifications (R1). S1 also examined the child nodes of LSTM and GRU. LSTM is extended to GridLSTM and GRU is extended to Lattice Recurrent Unit by dividing the data flow into two dimensions: time dimension (i.e., horizontal direction) and depth dimension (i.e., vertical direction), as shown in Fig. 7. By examining these evolutionary relationships, S1 learned that this architecture modification can be applied to improve the DNN architecture of interest.

6.1.2 Learning the Evolution Patterns of DNNs

S2 hoped to use DNN Genealogy to obtain an overview of the variety of DNN architectures developed for image classification, which is the basis of high-level applications such as semantic segmentation and video recognition.

Understanding the Evolution of Certain Types of Architectures (R2). By selecting “classification\ non-sequential data” in the control panel, S2 acquired the evolution of the relevant DNNs in the *Evolution View*. DNN Genealogy identified four types of architectures in these DNNs. Color was used to encode these different architecture types. The four architecture types are not independent but interacted with one another. Several diverging (Fig. 8A) and emerging (Fig. 8B) branches appeared. S2 used the control panel (Fig. 1A) to observe the evolution of the different types of architectures.

S2 first analyzed the evolution of *multi-branch*, which first appeared at Inception and diverged immediately after this DNN (Fig. 8a). While some child nodes of Inception have similar *multi-branch* architectures, others are the combination of *multi-branch* and other types of architectures. For example, ResNeXt (Fig. 8B) is the combination of *multi-branch* and *skip connections*.

S2 then observed the evolution of *skip connections*. Different from the evolution pattern of *multi-branch*, *skip connections* first appeared at HighwayNets but diverged only after ResNet (Fig. 8b). To find the reasons for this delay, S2 compared the network glyph of Highway Nets with that of ResNet. He observed that “ResNet removes the gate and directly adds the output of skip connections to the data (Fig. 8D)”. The gate, which learns parameters and is dependent on the input, decides whether the output of skip connections should affect the data. Therefore, compared with ResNet, Highway Nets has more parameters and is more complicated [62].

Comparing DNNs in the Context of Evolution (R6). S2 had a special interest in a region where the evolution of *skip connections* diverged and converged repeatedly, as shown in Fig. 8 C. He clicked and opened their network glyphs to compare their architectures (Fig. 8c) and immediately identified the difference between ResNet and DenseNet. First, in a building block, ResNet has only one skip connection (Fig. 8 F), whereas DenseNet has dense skip connections, which make one layer connect to all its previous layers. Second, ResNet combines branches through addition (\oplus , Fig. 8E) while DenseNet combines branches through concatenation (\parallel , Fig. 8G).

The major difference among DenseNet, MixNet, and DPN is how they combine branches, i.e., through addition, concatenation, or the combination of addition and concatenation. This difference made S2 wonder, “What is the difference between different types of combinations?”

S2 hoped to obtain answers from a detailed examination of the MixNet- and DPN-related papers. He then switched to the *Text View* and opened the papers of DPN and MixNet, from which he found the answers to his question: “Concatenation encourages new feature exploration but can lead to information redundancy. Addition enables feature reuse but may impede the information flow.”

6.2 Applying DNNs

This case study demonstrated that DNN Genealogy offered informative guidance in applying DNNs to real-world applications. We collaborated with a first-year PhD student (S3) and a teaching assistant in a computer vision course (S4).

6.2.1 Choosing a DNN

S3 was working on a project in which an object detection technique is needed. Because DNNs have achieved great success in computer vision, S3 decided to use a DNN for his project. This object detection needed to be carried out in a timely fashion on computationally limited devices, for example, a mobile phone. Thus, S3 hoped to use a lightweight DNN. In addition, S3, with limited experience in DNNs, preferred a simple DNN that was easy to understand and tune.

There are many available DNNs for object detection, such as Faster R-CNN [51], YOLO [50], and SSD [43]. S3 had little

idea on which one to choose. S3 also remembered that he had learned Faster R-CNN in a machine learning course. He wondered whether Faster R-CNN was more suitable for his project in comparison with other DNNs. Thus, S3 used DNN Genealogy to select an appropriate DNN for his project.

Finding State-of-the-art DNNs (R3). S3 first examined state-of-the-art DNNs using the *Performance View*, which provides three measures for comparing the performance of DNNs for object detection. Frames per second (FPS) measures the model efficiency. By contrast, the two other measures, PASCAL VOC [17] and Microsoft COCO [37], evaluate the model accuracy on the benchmark datasets. Hovering on the corresponding box plots, S3 found that the best performances of the three measures were achieved by three different DNNs. YOLO achieves the highest FPS, whereas DSOD [55] and FPN [36] achieves the highest accuracy on PASCAL VOC dataset and COCO dataset, respectively.

Comparing DNNs (R6). S3 then compared the three DNNs to select an appropriate one for his project. After examining the architectures using the network glyphs, he found that FPN had a more complicated architecture (Fig. 1 F) compared with the other two DNNs. FPN is a complex DNN architecture that includes a bottom-up pathway for reducing the spatial dimension, a bottom-up pathway for upsampling the coarse feature, and lateral connections for merging features from the bottom-up pathway and features from the top-down pathway. Preferring simpler architectures, S3 excluded FPN. Next, he compared YOLO and DSOD in the *Performance View* (Fig. 1b). While YOLO achieves the highest FPS, it has relatively low accuracy. Its accuracy on PASCAL VOC dataset is worse than 75 percent of the available DNNs, as its bar is lower than the lower quartile of the corresponding boxplot (Fig. 1 C). On the contrary, while DSOD has a lower FPS than YOLO, it is still better than 75 percent of the available DNNs, as its bar is above the upper quartile of the corresponding boxplot (Fig. 1 B). S3 concluded that DSOD made a better balance between efficiency and accuracy compared with YOLO. He finally decided to use DSOD in his project.

Analyze Relationships Among DNNs (R1). With little knowledge about DSOD, S3 was curious about how it outperforms Faster R-CNN, a DNN that he was familiar with. In the *Evolution View*, S3 observed that DSOD was connected with Faster R-CNN through SSD. He first read the annotation on the edge to examine the relationships between Faster R-CNN and SSD (Fig. 1 D). In comparison with Faster R-CNN, SSD has no object proposal network and can predict the bounding boxes and labels simultaneously. This difference leads to a simple architecture (simple network glyph), high efficiency (high FPS value), and high accuracy. By comparing the network glyphs and reading the edge annotation, S3 found that DSOD extends SSD by applying dense layer-wise connections (Fig. 1 E). Such dense layer-wise connections improve the model accuracy further. S3 said that the information provided by DNN Genealogy helped him better understand DSOD.

6.2.2 Deriving Guidance for Designing DNNs

S4 was preparing slides related to a course project, in which the students were asked to develop a DNN for the Kaggle Dogs versus Cats competition [2]. He hoped to provide design guidance to the students from comparing and

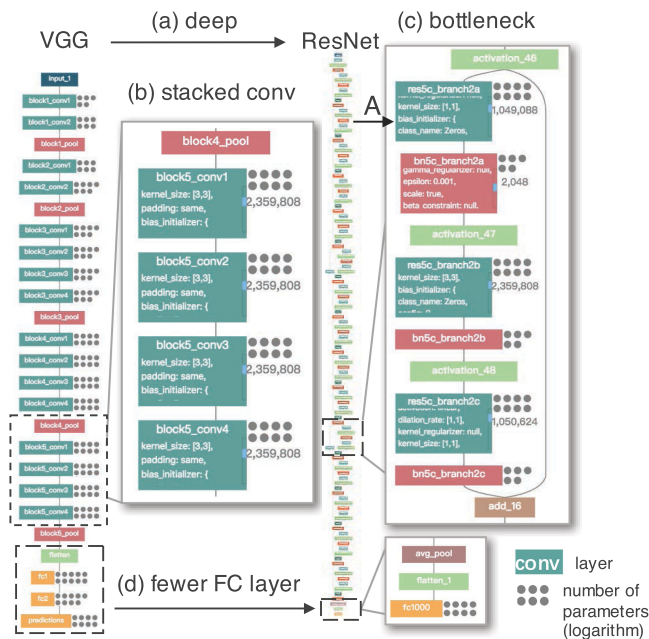


Fig. 9. Comparing the architecture of VGG with the that of ResNet. Four architecture modifications can be identified from the comparison: (a) A deeper architecture; (b) stacked convolution layers; (c) the bottleneck design; (d) fewer fully connected layers.

analyzing several representative DNNs. By default, DNN Genealogy shows the network glyphs of AlexNet, VGG, and ResNet because of their high importance scores. He then analyzed and compared these DNNs using DNN Genealogy.

Investigating DNNs from Different Aspects (R5). S4 first compared VGG and AlexNet. Through the *Performance View*, he found that VGG improves the accuracy of AlexNet at the cost of increasing model parameters. He then compared the detailed architecture of VGG and AlexNet. S4 immediately identified that VGG has a *deeper architecture* (i.e., more layers) than AlexNet. He also found that VGG has *stacked convolution layers* (Fig. 9b). In particular, VGG stacks convolution layers with a small filter size (3×3) to replace one convolution layer with large filter size (e.g., 11×11 , 5×5), which is used in AlexNet. This architecture modification increases the depth and the performances of a DNN. At the same time, convolutional layers with smaller filter size save computational cost. However, VGG still requires considerably more parameters than AlexNet because it has much more layers.

Next, S4 compared ResNet with VGG (Fig. 9). The *Performance View* showed that ResNet improves the accuracy and reduces the model parameters of VGG. He opened the *Architecture View* to examine how this was achieved through the architecture design. S4 found that ResNet has a *deeper architecture* (i.e., more layers) than VGG. Given that ResNet has more layers but fewer parameters than VGG, S4 deduced that ResNet must have fewer parameters each layer. He then examined the detailed architecture of ResNet and compared it with VGG. S4 focused on two types of layers that learn parameters, convolution layers and fully connected layers. He noticed that ResNet applies a design called *bottleneck* [23] (Fig. 9c). The bottleneck design is a stack of three convolution layers with the filter sizes of 1×1 , 3×3 , and 1×1 . The 1×1 layers are responsible for reducing/increasing dimensions so that the 3×3 layers only need to handle small input/output

dimensions. Scrolling down to the bottom of the architecture, S4 noticed that ResNet has *fewer fully connected layers* than VGG by adding a global average pooling layer (Fig. 9d). ResNet only has one fully connected layer, whereas VGG has three. Given that fully connected layers usually have a large number of parameters, this strategy significantly reduces the number of parameters of a DNN.

7 DISCUSSION

Generality. The methods developed in this work can be generalized to similar problems. First, in the development of an educational tool for a specific problem, representative papers can be obtained from the massive literature by adapting our semi-automatic four-step pipeline. Second, the system framework and the set of visual designs can be extended to other types of DNNs. For example, the genealogy of generative DNNs can be analyzed based on the same manner, namely, by presenting the performance and architecture of individual DNNs and by presenting the evolutionary relationships among different DNNs. The network glyph we proposed can be extended to generative DNNs.

Knowledge Learned. During the development and the case studies of DNN Genealogy, we have derived various knowledge about DNNs, including nine typical architecture types, a set of evolution patterns, and several beneficial architecture modifications. The derived knowledge demonstrates the educational value of DNN Genealogy. We believe that much knowledge still remains uncovered and anticipate that more knowledge will be derived by the users of DNN Genealogy. Meanwhile, DNN Genealogy keeps updated to include the new advances in DNNs. New knowledge will be generated and introduced from these updates.

Limitations & Future Work. Our case studies demonstrate the effectiveness and usefulness of DNN Genealogy. Nevertheless, a space for improvement remains.

First, DNN Genealogy mainly focuses on DNN architectures. Although the great success of DNNs is mainly due to the advances in architecture design [35], training methods should not be overlooked. An understanding of a variety of training methods helps users improve their understanding of the working mechanisms of DNNs and thus enable them to appropriately apply DNNs to their problems. The current version of DNN Genealogy provides a brief introduction to the commonly used training methods. In the future, we plan to visually connect the evolution of training methods with the evolution of DNN architectures. Such connections can help users determine which training method is appropriate for an architectures of interest.

The scope of this work is limited to DNNs for the three benchmark applications. In the future, we plan to extend DNN Genealogy by including more types of DNNs (e.g., deep generative models). We have added some newly proposed DNNs for non-Euclidean data (e.g., GraphSage [22], graph convolutional network [31]) in DNN Genealogy.

Third, we implement a heuristic DOI algorithm. Although this algorithm works well in practice according to our case studies, the importance score function considers only limited aspects of a DNN, namely, the number of citations and the public interest in this DNN based on Google Trends [2].

However, different users may define the importance of a DNN from different aspects, such as performance, the complexity of architecture, and relevance to a topic. We aim to address this issue by using attributes other than those used in the current work to describe a DNN and by introducing a set of interactions that enables users to formulate and modify the importance scores of DNNs.

8 CONCLUSION

In this work, we have presented DNN Genealogy, an interactive visualization tool that offers a visual summary of existing DNNs, to help in the understanding and application of DNNs. We conducted a systematic collection of the recent massive literature of DNNs and summarized 66 representative DNNs for benchmark applications. Nine types of architecture were identified through a detailed analysis of these DNNs. A set of visualizations was developed to help users learn the representative DNNs from various aspects, including architecture, performance, and evolutionary relationships. Two case studies were conducted to demonstrate the utility and usability of DNN Genealogy, especially in providing guidance for users in understanding, applying, and optimizing DNNs.

With the support of the open-source project (<https://github.com/wangqianwen0418/DNN-Genealogy>), we call for participation in the validation and extension of DNN Genealogy.

ACKNOWLEDGMENTS

This research was funded by National Key R&D Program of China (No. 2018YFB1004300), the National Natural Science Foundation of China (No.s 61761136020, 61672308), the HK TRS grant T41-709/17N, and the HK RGC GRF 16213317. We would like to thank all the participants involved in the studies for their valuable feedback, the reviewers for their constructive comments, and Dong Sun for proofreading the paper. We are grateful for the informative discussions with Furu Wei, Dong Chen, Lintao Zhang, Zichuan Lin, Tao Qin, Xin Tong, Xizhou Zhu, and Yang Liu from Microsoft Research, Asia.

REFERENCES

- [1] "Baidu Visual DL." 2017. [Online]. Available: <https://github.com/PaddlePaddle/VisualDL>, Accessed on: Mar. 28, 2018
- [2] "Google trends" 2006. [Online]. Available: <https://trends.google.com/trends/>, Accessed on: Mar. 28, 2018
- [3] "Kaggle Dogs vs. Cats." 2014. [Online]. Available: <https://www.kaggle.com/c/dogs-vs-cats>, accessed on: Sep. 28, 2018
- [4] "NVIDIA DIGITS: Interactive Deep Learning GPU Training System." 2016. [Online]. Available: <https://developer.nvidia.com/digits>, Accessed on: Mar. 28, 2018
- [5] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, et al., "Deep speech 2: End-to-end speech recognition in english and mandarin," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 173–182.
- [6] M. Asadi-Aghbolaghi, A. Clapes, M. Bellantonio, H. J. Escalante, V. Ponce-López, X. Baró, I. Guyon, S. Kasaei, and S. Escalera, "A survey on deep learning based approaches for action and gesture recognition in image sequences," in *Proc. 12th IEEE Int. Conf. Autom. Face Gesture Recognit.*, 2017, pp. 476–483.
- [7] J. Bae and B. Watson, "Developing and evaluating quilts for the depiction of large layered graphs," *IEEE Trans. Vis. Comput. Graph.*, vol. 17, no. 12, pp. 2268–2275, 2011.
- [8] A. Bilal, A. Jourabloo, M. Ye, X. Liu, and L. Ren, "Do convolutional neural networks learn class hierarchy?" *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 152–162, Jan. 2018.
- [9] R. A. Bjork, J. Dunlosky, and N. Kornell, "Self-regulated learning: Beliefs, techniques, and illusions," *Annu. Rev. Psychology*, vol. 64, pp. 417–444, 2013.
- [10] P. Chen, N. L. Zhang, L. K. Poon, and Z. Chen, "Progressive em for latent tree models and hierarchical topic detection," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 1498–1504.
- [11] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, "MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems," *arXiv:1512.01274*, 2015.
- [12] X. Chen and C. Lawrence Zitnick, "Mind's eye: A recurrent visual representation for image caption generation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 2422–2431.
- [13] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng, "Dual path networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2017, pp. 4470–4478.
- [14] F. Chollet, et al., "Keras," 2015. [Online]. Available: <https://github.com/keras-team/keras>, Accessed on: Mar. 28, 2018
- [15] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1251–1258.
- [16] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 933–941.
- [17] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.
- [18] G. W. Furnas, "Generalized fisheye views," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 1986, pp. 16–23.
- [19] E. R. Gansner, E. Koutsofios, S. C. North, and K.-P. Vo, "A technique for drawing directed graphs," *IEEE Trans. Softw. Eng.*, vol. 19, no. 3, pp. 214–230, Mar. 1993.
- [20] A. Garcia-Garcia, S. Orts-Escobedo, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, "A review on deep learning techniques applied to semantic segmentation," *arXiv:1704.06857*, 2017.
- [21] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, L. Wang, G. Wang et al., "Recent advances in convolutional neural networks," *arXiv:1512.07108*, 2015.
- [22] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Advances Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.
- [23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
- [24] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, 2012.
- [25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [26] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv:1704.04861*, 2017.
- [27] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, vol. 1, Art. no. 3.
- [28] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 646–661.
- [29] M. Kahng, P. Y. Andrews, A. Kalro, and D. H. Chau, "ActiVis: Visual exploration of industry-scale deep neural network models," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 88–97, Jan. 2018.
- [30] N. W. Kim, S. K. Card, and J. Heer, "Tracing genealogical data with timenets," in *Proc. Int. Conf. Adv. Visual Interfaces*, 2010, pp. 241–248.
- [31] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv:1609.02907*, 2016.
- [32] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork RNN," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1863–1871.
- [33] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
- [34] G. Larsson, M. Maire, and G. Shakhnarovich, "FractalNet: Ultra-deep neural networks without residuals," in *Proc. Int. Conf. Learn. Representations*, 2017.
- [35] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

- [36] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 936–944.
- [37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
- [38] D. Liu, W. Cui, K. Jin, Y. Guo, and H. Qu, "Deeptracker: Visualizing the training process of convolutional neural networks," *ACM Trans. Intell. Syst. Technol.*, vol. 10, 2018, Art. no. 6.
- [39] M. Liu, J. Shi, K. Cao, J. Zhu, and S. Liu, "Analyzing the training processes of deep generative models," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 77–87, Jan. 2018.
- [40] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards better analysis of deep convolutional neural networks," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 91–100, Jan. 2017.
- [41] S. Liu, X. Wang, C. Collins, W. Dou, F. Ouyang, M. El-Assady, L. Jiang, and D. Keim, "Bridging text visualization and mining: A task-driven survey," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 7, pp. 2482–2504, Jul. 2019.
- [42] T. Liu, N. L. Zhang, and P. Chen, "Hierarchical latent tree analysis for topic detection," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2014, pp. 256–272.
- [43] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
- [44] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," arXiv:1508.04025, 2015.
- [45] M. McGuffin and R. Balakrishnan, "Interactive visualization of genealogical graphs," in *Proc. IEEE Symp. Inf. Vis.*, 2005, pp. 16–23.
- [46] Y. Miao, M. Gowayyed, and F. Metze, "EESSEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Proc. Workshop Autom. Speech Recognit. Understanding*, 2015, pp. 167–174.
- [47] Y. Ming, S. Cao, R. Zhang, Z. Li, Y. Chen, Y. Song, and H. Qu, "Understanding hidden memories of recurrent neural networks," arXiv:1710.10777, 2017.
- [48] N. Pezzotti, T. Hillt, J. V. Gemert, B. P. F. Lelieveldt, E. Eisemann, and A. Vilanova, "Deepees: Progressive visual analytics for designing deep neural networks," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 98–108, Jan. 2018.
- [49] P. E. Rauber, S. G. Fadel, A. X. Falcao, and A. C. Telea, "Visualizing the hidden activity of artificial neural networks," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 101–110, Jan. 2017.
- [50] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.
- [51] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2015, pp. 91–99.
- [52] D. Sacha, L. Zhang, M. Sedlmair, J. A. Lee, J. Peltonen, D. Weiskopf, S. C. North, and D. A. Keim, "Visual interaction with dimensionality reduction: A structured literature analysis," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 241–250, Jan. 2017.
- [53] H. Salehinejad, J. Baarbe, S. Sankar, J. Barfett, E. Colak, and S. Valaee, "Recent advances in recurrent neural networks," arXiv:1801.01078, 2017.
- [54] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.
- [55] Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, and X. Xue, "DSOD: Learning deeply supervised object detectors from scratch," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, vol. 3, Art. no. 7.
- [56] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015.
- [57] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush, "LSTMVIS: A tool for visual analysis of hidden state dynamics in recurrent neural networks," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 667–676, Jan. 2018.
- [58] K. Sugiyama, S. Tagawa, and M. Toda, "Methods for visual understanding of hierarchical system structures," *IEEE Trans. Syst. Man Cybern.*, vol. 11, no. 2, pp. 109–125, Feb. 1981.
- [59] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 4278–4284.
- [60] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," arXiv:1503.00075, 2015.
- [61] F. Van Ham and A. Perer, "search, show context, expand on demand: Supporting large graph exploration with degree-of-interest," *IEEE Trans. Vis. Comput. Graph.*, vol. 15, no. 6, pp. 953–960, Nov./Dec. 2009.
- [62] A. Veit, M. J. Wilber, and S. Belongie, "Residual networks behave like ensembles of relatively shallow networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2016, pp. 550–558.
- [63] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3156–3164.
- [64] K. Wongsuphasawat, D. Smilkov, J. Wexler, J. Wilson, D. Mané, D. Fritz, D. Krishnan, F. B. Viégas, and M. Wattenberg, "Visualizing dataflow graphs of deep learning models in tensorflow," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 1–12, Jan. 2018.
- [65] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," arXiv:1609.08144, 2016.
- [66] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5987–5995.
- [67] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," arXiv:1708.02709, 2017.
- [68] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," arXiv:1801.07883, 2018.
- [69] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2016.



Qianwen Wang received the BS degree from Xi'an Jiaotong University. She is now working toward the PhD degree at the Hong Kong University of Science and Technology. Her work strives to facilitate the communication between humans and machine learning models through creating interactive visual analysis systems. Her research interests include interactive machine learning, explainable artificial intelligence, and narrative visualization.



Jun Yuan is currently working toward the graduate degree at Tsinghua University. His research interests include visual analytics and explainable artificial intelligence.

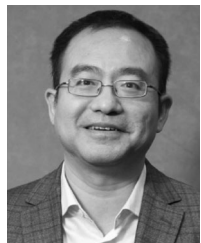


Shuxin Chen is currently working toward the dergraduate degree at Tsinghua University. His research interests include visual analytics and algorithm design.



Hang Su received the BS, MS, and PhD degrees from Shanghai Jiaotong University, China. He is an assistant professor with the Department of Computer Science and Technology at Tsinghua University. His research interests include the development of computer vision and machine learning algorithms for solving scientific and engineering problems arising from artificial learning, reasoning, and decision-making in the high-dimensional and dynamic worlds. His current work involves both the foundations of interpretable

machine learning and the applications of image/video analysis, based on which he has published around 50 papers including CVPR, ECCV, TMI, etc. He has served as senior PC or PC members in the dominant international conferences. He is a member of the IEEE.



Huamin Qu received the BS degree in mathematics from Xi'an Jiaotong University, China, and the MS and PhD degrees in computer science from the Stony Brook University. He is a professor with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His main research interests include visualization and human-computer interaction, with focuses on urban informatics, social network analysis, e-learning, text visualization, and explainable artificial intelligence. For more information, please visit <http://www.huamin.org>.



Shixia Liu received the BS and MS degrees from the Harbin Institute of Technology, and the PhD degree from Tsinghua University. She is an associate professor with Tsinghua University. Her research interests include visual text analytics, visual social analytics, interactive machine learning, and text mining. She worked as a research staff member at IBM China Research Lab and a lead researcher at Microsoft Research Asia. She is an associate editor of IEEE TVCG and Information Visualization.

▷ **For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.**