博士論文

Using Deep Learning to Make Data Manifest Knowledge

(データに知識を説明させるための深層学習活用)

張確軒

Using Deep Learning to Make Data Manifest Knowledge

by

Quexuan Zhang

Submitted to the Department of Systems Innovation, School of Engineering
on December 1, 2020, in partial fulfillment of the
requirements for the degree of
Doctor of Engineering

Abstract

In the Industry 4.0 and Society 5.0 concepts, data is still the core component for the highly expected technologies, such as cyber-physical systems, the internet of things, artificial intelligence, and big data analytics. Deep learning (DL) has been widely used as a versatile and high-performance tool to extract useful knowledge and information from data for problem-solving in various domains. By participating in the IMDJ workshops aiming at better data utilization, we found that people criticize DL's poor interpretability. However, some data is difficult to extract features or label targets by manual work, such as Fourier speckle patterns captured from the laser machining process. Thus, DL is still a potential solution for those complicated problems. Motivated by these requirements, the following methods were proposed to serve a DL-based knowledge discovery framework.

To enhance the interpretability of deep models, we proposed two model interpretation methods, nonlinearized relevance propagation (NRP) and key input subset sampling (KISS), to explain deep models by visualizing the relation between input and output. NRP is altered from layer-wise relevance propagation (LRP) by introducing nonlinear functions into the relevance decomposition and performed better than LRP in the experiment on a question answering model. KISS overlooks all the relations between the input and each candidate answer based on the energy-based model theory with only forward information or the combination of forward and backward information from the model. In the experiment on the image classification models, KISS outperformed other contrastive models. In the case study, we evaluated the performance of different deep models in supervised single-task and multi-task learning (MTL) on the laser machining data and found that the AlexNet-in-MTL model performed better than the other models. However, it is still challenging to apply supervised DL approaches to sequential data where anomalies are hard to be separated. Thus, I proposed pessimistic contrastive learning (PCL) to drive data points to compare with each other in a sequence to predict the anomalies. In the experiments, PCL was compared to two commonly used anomaly detection methods on one-dimensional synthetic data and then was applied to find illegible handwritten digits from the MNIST dataset. The evidence showed that PCL could give meaningful results for

anomaly detection.

Thesis Supervisor: Yukio Ohsawa
Title: Professor

Acknowledgments

I would first like to thank my supervisor, Professor Yukio Ohsawa, whose expertise broadened my perspective on innovation by combining technology and humans. He helps me consider the pain points in technologies and our life to find out new solutions. I genuinely appreciate his trust, patience, and continuous support.

I would like to acknowledge my colleagues in Ohsawa Lab. and The University of Tokyo for their excellent guidance and wonderful collaboration. Also, I would like to thank the Japan Science and Technology Agency (JST), and the Ministry of Education, Culture, Sports, Science and Technology (MEXT) for supporting part of this research.

In addition, I would like to thank my family for their unconditional support for my overseas study. You are always the lighthouse at the warmest harbor in the world for me. Finally, there are my friends, who are the source of great encouragement to me whether I am feeling depressed or cheerful.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Data becomes more and more important in knowledge and information exchange. Deep learning, which is a powerful and versatile technology to extract useful information from data for various tasks, is applied to the support of knowledge discovery in this work.

Chapter Two describes two proposed methods of model interpretation for deep models.

Chapter Three describes the case study on laser machining data.

Chapter Four describes a new deep learning method for anomaly detection on sequential data.

Chapter Five gives the conclusions and future works.

1.1 Background

First of all, let me introduce the importance of data for the industry and our life and present related deep learning methods for data utilization in brief.

1.1.1 Data for Us

Since the Digital Revolution began, data has been playing a significant role as a purveyor of information. For better continuous developing and problem-solving in

industry and society, *the Fourth Industrial Revolution* (or Industry 4.0) [1] and Society 5.0 were proposed [2]. Industry 4.0 focuses on intelligent technology for the automation of industrial practices, while Society 5.0 aims at using new technologies to improve the environment of our life.

One of the core technologies in these blueprints is the cyber-physical system (CPS). In CPS, physical mechanisms are controlled and monitored by computers via networks and algorithms. By analyzing the information collected from physical processes, a virtual version of the physical world can be modeled and designed. Associated technologies of CPS include internet of things (IoT), cloud computing, artificial intelligence (AI), and big data analytics. Data is used for information extraction, information transfer, model learning, signal analysis, and the like in these technologies. Without data utilization, it would be hard to do innovations for the industry and our society.

1.1.2 Knowledge from Data and Deep Learning

Knowledge discovery in database (KDD) is the process of extracting useful information from data to solve problems [3]. Machine learning (ML) is one of the thriving technologies for knowledge discovery and data mining. Recently, Deep learning (DL), a study branch of ML, attracts worldwide attention due to its outperformance on many computational tasks in different domains [4–9]. One reason for the high performance is that DL is good at extracting representative features from data for tasks' objectives. Nevertheless, DL methods use multiple neural network layers, which can approximate any functions theoretically [10], to learn information for prediction from the input data. Thus, it is difficult for humans to understand the inference process of those "black-box" models. Besides, it is a fact that most of the outperformed deep models are based on supervised learning that needs massive human-labeled training data for high accuracy. Furthermore, the large number of parameters in deep models usually leads to an overfitting problem.

Fortunately, many researchers are working hard on those issues in DL. Model interpretation (MI) is to explain the prediction processes of machine learning models for

humans, in which some methods have been proposed for deep model explanation [11, 12]. Beyond supervised learning, unsupervised or self-supervised approaches aim to reduce manual work for data labeling [13, 14]. To conquer the overfitting problem, applying Dropout, data augmentation, or multi-task learning (MTL) with parameter sharing could improve DL models' generalization [15–17].

Thanks to the mentioned endeavors, DL can be not only a kind of fitting tool but also a potential application for knowledge extraction. Section 1.3 will give a framework on how to make data manifest knowledge by using DL.

1.2 Motivations

Due to the complications in actual practices, people need to do decision-making on the data utilization for better knowledge discovery. Innovators Marketplace on Data Jackets (IMDJ) [18] is a game-style platform to support decision-making. This work is also strongly motivated by IMDJ.

In IMDJ, participants create new requirements and solutions by combining Data Jackets (DJs) and Tool Jackets (TJs). In this process, they also negotiate the "prices" of solutions to discuss the value of data. Here, DJs or TJs keep the summaries of different datasets or techniques in data utilization, which the data holders or domain experts provide. Besides, the relationships between the jackets are revealed by a visualization tool (*e.g.*, KeyGraph [19]) to facilitate the discussion on cross-disciplinary databases or analytical methods. Those data or tools generated from the solutions can be used for the next IMDJ toward the spiral of innovation.

By participating in IMDJ workshops, I know that people are not satisfied with non-deep analytical techniques, but either the interpretability of neural networks. Non-deep methods usually require much human work into feature engineering to improve performance. On the other hand, deep models can automatically extract representative features from data and produce even better results. However, the black-box and overfitting traits could confuse users on whether they should trust the deep models. Therefore, MI becomes one of our research topics described in Chapter 2.

Sometimes, it may happen that even the domain experts do not comprehend their datasets well. There was an IMDJ workshop held for data scientists and physicists to solve problems on laser machining. The physicists collect photograph data in the machining process, which records the speckle patterns captured on the Fourier plane [20]. They expect to maximize the value of data in applying the data to various downstream tasks. However, in the face of such micro-scale data generated by complex procedures, it is hard to perform feature engineering or label anomalies. As a consequence, DL approaches with high generalization are required. For this sake, we analyzed a set of the laser machining data and evaluated the performance of different deep models on it as reported in Chapter 3, and then developed a non-supervised DL-based anomaly detection method as presented in Chapter 4.

1.3 Deep Learning for Knowledge Discovery

According to the background and motivations mentioned above, it is worth using deep learning to support discover and apply knowledge hidden behind data. Here, I give a DL-based knowledge discovery framework that was inspired by IMDJ's visualization and the spiral of innovation:

1. We employ deep learning algorithms on the target data and select the models with high accuracy and generalization.

2. We apply MI approaches on the selected deep models and visualize the explanatory result for human users.

3. It is expected that the users obtain new knowledge via combining the explanation and their prior knowledge.

4. The new knowledge could help us improve the algorithms for better models.

In this thesis, knowledge is defined as useful information for problem-solving, including two aspects: 1) representative features extracted by deep models from data, and 2) the novelty discovered by humans combining their prior knowledge and the

meaningful information from data. MI could help people understand what deep models have learned or consider their reliability and improvement. Also, the discovery could help better problem-solving.

This thesis focuses on two proposed MI methods for DL and a multi-task self-supervised approach for anomaly detection (AD). The MI methods include model-agnostic (Section 2.2), model-specific (Section 2.3.2), and our original hybrid styles (Section 2.3.4). As AD could help people find novelty or change points in data, it is a significant study subject for knowledge discovery [21]. I designed novel objective functions (Section 4.2.1), a contextual information-based neural network model (Section 4.2.2), and brand-new algorithms (Section 4.3) for the training process of AD tasks.

1.4 Other Deep Learning Applications on Knowledge Discovery

There have been some DL approaches proposed for knowledge discovery in different domains. Rather *et al.* used the word2vec model to extract knowledge from biomedical textual data for novelty discovery on biomedicine [22]. Huang *et al.* trained deep belief networks in multi-task learning to monitor and control power grids on power system security assessment [23]. Xu *et al.* applied deep learning to automatically extract entities from encyclopedia articles into the knowledge base [24]. In these applications, people were leveraging DL's powerful feature extraction ability and combining with human knowledge to discover novelty, yet hardly considering the interpretability of deep models.

Chapter 2

Model Interpretation for Deep Models

Dissimilar to explainable machine learning models such as LASSO [25] and decision trees [26], it is hard to explain the prediction process in deep models due to the complicated internal nonlinearity. Especially for the applications related to life and property, *e.g.*, healthcare and autonomous vehicles, it is vital to require explanations for the models [27, 28].

Given a data sample $\boldsymbol{x} = [x_1, \ldots, x_n] \in \mathcal{X}$ as input, where each element $x_i \in \mathbb{R}^d$ is an d-dimensional element of the input, and a trained deep model $f : \mathcal{X} \mapsto \mathbb{R}^c$, a practical way of interpreting the model f is to visualize importance scores $\boldsymbol{s} = [s_1, \ldots, s_n]$ for each element x_i with respect to the output $f(\boldsymbol{x})$. With the visualization, people could seize meaningful information extracted from data by the models intuitively.

In this chapter, two proposed MI methods, nonlinearized relevance propagation (NRP) and key input subset sampling (KISS), will be introduced in the following.

2.1 Taxonomy

We can typically group MI approaches into two categories, model-agnostic and model-specific [29]. We can use this taxonomy to categorize MI methods for deep models as

well.

2.1.1 Model-agnostic

Model-agnostic methods test the relevancy between input and output in a forward style without model architecture information. We can continue to group them into three subgroups as follows:

- **Explainable model-based methods**: explainable models are trained to locally fit the prediction of the target model, such as LIME [30], KernelSHAP [31], and Soft decision trees [32].
- **Difference-based methods**: the difference related to the output by removing input elements are measured as the importance scores, such as Occlusion [33], DeepVis [34], and SIScollection [35].
- **Neural-network-based methods** are by training extra simple neural networks to model the problem, *e.g.*, L2X [36].

2.1.2 Model-specific

Model-specific methods must use the model architecture to propagate the relevancy from output to input backward. Also, they can be categorized into three subcategories as follows:

- **Gradient-based methods**: gradients of output with respect to each input element are computed as the importance scores by the chain rule, *e.g.*, saliency maps [37, 38].
- **Parameter-based methods**: importance weights from the output are propagated to the corresponding input layer-by-layer, such as LRP [39] and DeepLIFT [40].
- **Inverse-based methods**: inverse architectures to the original models are designed to reverse the operations, *e.g.*, Deconvolution [33].

NRP is a model-specific method, and KISS can be a model-agnostic method or a hybrid method combining forward and backward information.

2.2 Nonlinearized Relevance Propagation

This method is a model-specific approach to solve the problems in the prior work layer-wise relevance propagation (LRP) [39] by introducing nonlinear functions in the propagation rules.

2.2.1 Prior Work: Layer-wise Relevance Propagation

LRP is a Taylor decomposition-based approach in which the output is resolved into relevance scores, and then the scores are backward propagated to the inputs layer-by-layer conservatively. More concretely, between a layer l and the successive one $l+1$ in a forward neural network, LRP keeps the consistency of relevance over layers by

$$\sum_j R_{j\leftarrow k}^{(l)} = R_k^{(l+1)} \tag{2.1}$$

and

$$R_j^{(l)} = \sum_k R_{j\leftarrow k}^{(l+1)}, \tag{2.2}$$

where j and k are the indices for neurons of the two layers, $R_{j\leftarrow k}$ denotes the propagated relevance score from neuron k to neuron j if they have a connection. Because of $\sum_j R_j^{(l)} = \sum_j \sum_k R_{j\leftarrow k}^{(l+1)} = \sum_k \sum_j R_{j\leftarrow k}^{(l)} = \sum_k R_k^{(l+1)}$, the relevance propagation in LRP meets the conservation property as this:

$$\sum_{i=1}^{d} R_i^{(0)} = \cdots = \sum_j R_j^{(l)} = \sum_k R_k^{(l+1)} = \cdots = f(\boldsymbol{x}). \tag{2.3}$$

Given a typical neural network layer as

$$\begin{aligned} c_{jk} &= a_j w_{jk} \\ c_k &= \sum_j c_{jk} + b_k \\ a_k &= \sigma\left(c_k\right), \end{aligned} \tag{2.4}$$

where a_j indicates the value of neuron j at layer l, c_k denotes a linear connection from all neurons at layer l to neuron k at layer $l+1$ with weights w_{jk} and a bias b_k, and σ is a nonlinear activation function.

In LRP, the decomposition rule for layer $l+1$ to layer l is given by

$$R_{j\leftarrow k} = \left[\alpha\left(\frac{c_{jk}^{+}}{\sum_j c_{jk}^{+} + b_k^{+}}\right) + \beta\left(\frac{c_{jk}^{-}}{\sum_j c_{jk}^{-} + b_k^{-}}\right)\right] R_k. \tag{2.5}$$

, where $(\cdot)^+$ and $(\cdot)^-$ denote the positive and negative parts respectively, and the parameters α and β satisfy $\alpha + \beta = 1$ and $\alpha > 0$ for the conservation. A practical choice of the parameters is $\alpha = 1, \beta = 0$ or $\alpha = 2, \beta = -1$. This rule is also called the $\alpha\beta$-rule.

Nevertheless, with the development of deep learning methods, many special layers rather than the typical ones have been designed. Since those special layers can not be characterized as Equation 2.4 [41], some rules were proposed to adapt the LRP framework as follows.

LRP for Gates

Gates are applied to regulate the forward information transfer in recurrent neural networks (RNNs), such as LSTM [42] and GRU [43]. A gate weight w_j^g is often calculated by Equation 2.4 with the sigmoid activation, and the layer is formulated by $a_k = w_j^g \cdot a_j$.

Arras *et al.*employed LRP to explain LSTM-based text sentiment classification [44]. Since gates can be considered as the probability of the information passing by them,

the relevance is fully transferred for the conservation as

$$R_{j\leftarrow k} = R_k. \tag{2.6}$$

LRP for Pooling

The pooling, such as max-pooling and avg-pooling, is applied in lots of deep models. Avg-pooling takes the middle point of specific neurons to aggregate the features, while max-pooling gives the maximum value in a region of neurons for the approximate invariant.

A possible rule for avg-pooling is similar to the one for element-wise above as

$$R_{j\leftarrow k} = \frac{a_j}{\sum_j a'_j} R_k. \tag{2.7}$$

Besides, the propagation rule for max-pooling applied by Ding *et al.* [45] was

$$R_{j\leftarrow k} = \begin{cases} R_k & \text{if } j = \max\limits_j \{a_j\} \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

Notwithstanding, Equation 2.7 and 2.8 are not adaptive with $\alpha\beta$-parameters. Especially, in Equation 2.8, only the maximum neuron at layer l fully captures the relevance from the neuron at layer $l+1$. However, there is much information from the other neurons at layer l for the comparison in the max function. Using Equation 2.8 could lose lots of contrastive information for those neurons in the monopoly of relevance.

LRP for Hadamard product

Hadamard product, a.k.a. element-wise multiplication, is involved in LSTM, GRU, and cosine similarity units. A Hadamard product layer is given by $a_k^{(l+1)} = {a_j}^{(l)} \cdot {a'_j}^{(l)}$.

Ding *et al.*applied LRP to analyze a GRU-based sequence-to-sequence model for neural machine translation [45], where the relevance propagation rule for a Hadamard

product layer was

$$R_{j\leftarrow k} = \frac{a_j}{a_j + a'_j} R_k. \tag{2.9}$$

However, this rule breaks the conservation property when a_j and a'_j have different signs. Also, it is not applied with the $\alpha\beta$-parameters.

2.2.2 Proposed $\alpha\beta$-rules

To supplement LRP, we proposed $\alpha\beta$ rules for pooling-weight layers [46]. There were three reasons why we considered nonlinear functions as follows:

1. **Rules for special layers.** As mentioned in Section 2.2.1, there are various layers designed for deep models. The relevance decomposition rules for those layers are created in different applications. Those rules which do not satisfy the Taylor assumption could produce relevance errors in the propagation process. One example is the attention pooling network (APN) [47]. This model uses a soft alignment [48] and max-pooling to compute pooling weights, in which there could be an information loss problem by using Equation 2.8 and 2.5.

2. **Error accumulation.** If the neurons at a deeper layer receive wrong relevance produced by the improper rules, the errors could accumulate layer-by-layer to the neurons at shallower layers. This problem could be prominent in very deep models (*e.g.*, RNN-based networks).

3. **Adjustment effect.** An appropriate nonlinear function could amend the relevance distribution to a more meaningful value space. We expected that this property could help inputs receive constructive information from the model outputs.

Improved $\alpha\beta$-rule for Pooling-weighted Layers

To solve the problem mentioned above, we introduced nonlinear functions h for pooling-weighted layers as follows.

Definition 1 ($\alpha\beta$-rule for pooling-weighted layers)**.** *For neurons at a pooling-weighted layer decomposing relevance R_k into $R_{j\leftarrow k}$ to neurons in the last layer,*

$$R_{j\leftarrow k} = \left[\alpha\left(\frac{h\left(c_{jk}^{+}\right)}{\sum_j h\left(c_{jk}^{+}\right)}\right) + \beta\left(\frac{h\left(c_{jk}^{-}\right)}{\sum_j h\left(c_{jk}^{-}\right)}\right)\right] R_k, \tag{2.10}$$

where $\alpha + \beta = 1$ and $\alpha > 0$.

We let the nonlinearization h keep the monotonicity of the $(\cdot)^+$ and $(\cdot)^-$ parts, *e.g.*, hyperbolic tangent $\tanh x$. We can also choose positive semidefinite axial symmetry functions, such as $h(x) = x^2$. Furthermore, we can compose monotonic functions and the absolute value function as the nonlinearizer, *e.g.*, $h(x) = \sqrt{|x|}$ or $h(x) = \log(|x|+1)$. Besides, we construct value masks to avoid unexpected relevance to the zero-neurons, *i.e.*, $h(0) = 0$, when employing nonlinear functions shifting over the origin, such as $h(x) = \exp(|x|)$, $h(x) = \text{sigmoid}(|x|)$ and hyperbolic cosine $\cosh(x)$.

$\alpha\beta$-rule for Hadamard Product

We found that the decomposition rule given by Equation 2.9 did not conserve the relevance scores if the signs of the two elements were different. Therefore, we provided a conservational version as follows.

Definition 2 ($\alpha\beta$-rule for Hadamard product)**.** *For neurons at a Hadamard product layer decomposing relevance R_k into $R_{j\leftarrow k}$ to neurons in the last layer,*

$$R_{j\leftarrow k} = \begin{cases} \dfrac{a_j}{a_j + a_j'} R_k & \text{if } a_j > 0, a_j' > 0 \text{ or } a_j < 0, a_j' < 0 \\ \alpha R_k & \text{for } a_k \leftarrow a_j \text{ if } a_j > 0, a_j' < 0 \\ \beta R_k & \text{for } a_k \leftarrow a_j \text{ if } a_j < 0, a_j' > 0 \\ 0 & \text{if } a_i = 0 \text{ or } a_j = 0 \end{cases} . \tag{2.11}$$

2.2.3 Application to Attentive Pooling Network

In Section 2.2.4, we will report the experiment result of applying NRP to the attentive pooling network (APN) [47]. APN is a bidirected LSTM-based model with the

max-pooling-based attention mechanism and cosine similarity for question answering.

The architecture of APN is shown on the left of Figure 2-1. Also, the right of Figure 2-1 shows how we employed NRP to APN.

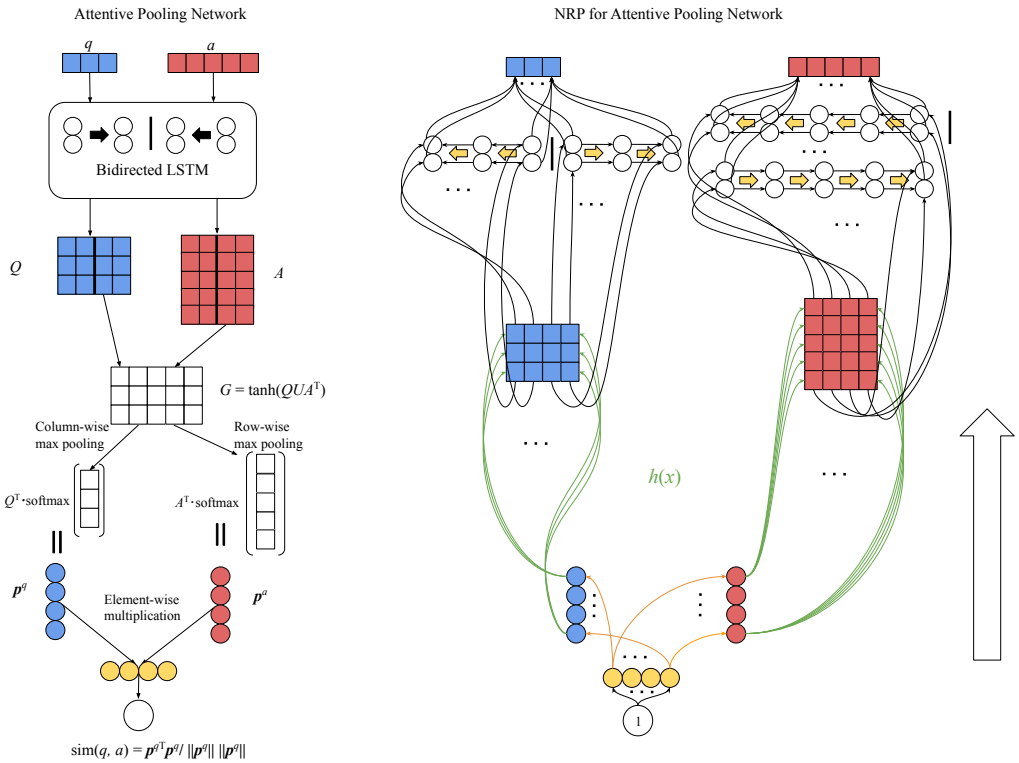


Figure 2-1: Architecture of APN and NRP application to APN. In the left figure, q is an input question, and a is an input answer. They are encoded into Q and A by the bidirected LSTM and weighted by column-wise or row-wise max-pooling of the attention G to produce projections p^q and p^a respectively. Then, the output is the cosine similarity of p^q and p^a. The better answers should have the higher similarities. In the right one, green connections denote where to apply Equation 2.10, while orange links are the places to use 2.11. For the relevance propagation of gates in LSTM, we still utilized the rule given by Equation 2.6

.

2.2.4 Experiments

This section reports the details of experiments and the results by comparing our method to saliency maps [37] and the original linear LRP.

Dataset and Preprocessing

In our experiments, we used nfL6[1] as the target dataset. The dataset contains 87,361 questions in 21 categories. Each question has a selected best answer along with one or more than one other answer. However, the best answer may have a duplicate in the other answer list for the same questions, and accordingly, we removed them in the preprocessing. We split each content into tokens by using spaCy[2] and cleaned up the URLs in the sentences.

Model Setup

Rather than using the whole dataset to train one model, we picked up three categories, Computer & Internet, Health and Society & Culture, as three subsets to train three different models. We divided them into training sets, development sets, and test sets by 8:1:1 in the original question orders.

We implemented APN in PyTorch [49] and trained three models for each subset with Adam optimizers [50]. We used 100-dimensional Glove embeddings [51] for the word representation and fine-tuned them during the training. Also, we set the patience to 9 epochs for the early stopping. Table B.1 shows the other hyper-parameters and the result states of selected models. Furthermore, the accuracy in the model evaluation was calculated by

$$ACC = \frac{|\{q|\mathrm{sim}(q, a_{\mathrm{best}}) > \max_{\boldsymbol{a}_{\mathrm{other}}}[\mathrm{sim}(q, a_{\mathrm{other}})]\}|}{|\boldsymbol{q}|}. \tag{2.12}$$

Results

We employed the token deleting test [41] to compare our method with SA and LRP in the linear setting. Deleting a token indicates setting the token's embedding to a zero vector. The algorithm of the test on a question-answer pair is shown in Algorithm 1. Besides, we chose the ground true best answers with token lengths greater than 9 for the experiments.

[1] https://ciir.cs.umass.edu/downloads/nfL6/
[2] https://spacy.io

Algorithm 1: Token deleting test

Input: a question q, an answer $a = [\boldsymbol{x}_i], 1 \leq i \leq L$, the maximum number of token deleted K, a trained APN model f, ordered relevance $\boldsymbol{r} = [r_j]$.

sort the tokens $[\boldsymbol{x}_i]$ into $\left[\boldsymbol{x}'_j\right]$ by the order of $\boldsymbol{r}$

for $j \leftarrow 1$ **to** K **do**

  $\boldsymbol{x}_i \leftarrow \mathbf{0}$ where $\boldsymbol{x}_i = \boldsymbol{x}'_j$

  record the model output $f(q, a)$

For the comparison of performance, we operated two experiments on the development sets and test sets. Experiment 1 observed the accuracy *reduction* on true positive (TP) samples while deleting tokens in answers by the *descending* order of the relevancies. Experiment 2 observed the accuracy *increment* on false positive (FP) samples while deleting tokens in answers by the *ascending* order of the relevancies. Table 2.1 and Table 2.2 show the results at deleting one token and five tokens by different methods and parameters for each data subset.

According to the results, we found that applying certain nonlinear functions to LRP helped the explainer capture more important inputs than SA and the linear setting. The performance for Society & Culture was especially remarkable in Experiment 1, and all the tests with $\alpha = 2$ were far better than SA. However, not all nonlinearizers had good results. Figure 2-2 shows the coefficients' distributions to decompose relevance scores by applying Equation 2.10 upon the best answers. Since the neurons' values were near zero in the APN models, the linear setting, hyperbolic tangent, and $h(x) = \log(|x| + 1)$ obtained similar results. The range of the values by square function was wider than the linear setting so that the relevance propagation could be too scattered. The rest of the functions produced more narrow but smooth distributions, which could help allocate the relevance information meaningfully according to Table 2.1 and Table 2.2.

However, for other special layers than the pooling-weight layer, appropriate selections of nonlinear functions are still unknown. A possible approach is to choose a small data subset and launch the tests to select the functions with high performance.

Table 2.1: Results of Experiment 1. Lower is better in a column.

Dataset		*Computer&Internet*		*Health*		*Society&Culture*	
ACC@#(tokens deleted)		1	5	1	5	1	5
SA		0.83275	0.50523	0.90131	0.61732	0.89944	0.80074
$h(x) = x$	$\alpha = 1$	0.83275	0.54495	0.89829	0.58912	0.89758	0.69088
$h(x) = \tanh(x)$	$\alpha = 1$	0.83345	0.54495	0.89829	0.58912	0.89758	0.69088
$h(x) = x^2$	$\alpha = 1$	0.84948	0.58885	0.90836	0.63545	0.89758	0.67225
$h(x) = \log(\lvert x\rvert + 1)$	$\alpha = 1$	0.83345	0.54425	0.89829	0.59013	0.89758	0.68901
$h(x) = \sqrt{\lvert x\rvert}$	$\alpha = 1$	0.83136	0.53659	0.89728	**0.55690**	0.86778	0.61266
$h(x) = \exp(\lvert x\rvert)$	$\alpha = 1$	**0.82927**	**0.49617**	0.89527	0.56999	0.81192	**0.34451**
$h(x) = \cosh(x)$	$\alpha = 1$	**0.82927**	**0.49617**	0.89527	0.56999	**0.81006**	**0.34451**
$h(x) = \text{sigmoid}(\lvert x\rvert)$	$\alpha = 1$	**0.82927**	**0.49617**	0.89527	0.56999	0.81192	**0.34451**
$h(x) = x$	$\alpha = 2$	0.95679	0.87944	0.89527	0.67372	0.88827	0.69646
$h(x) = \tanh(x)$	$\alpha = 2$	0.95679	0.87944	0.89527	0.67372	0.88827	0.69832
$h(x) = x^2$	$\alpha = 2$	0.95889	0.86969	0.89728	0.65962	0.87896	0.69460
$h(x) = \log(\lvert x\rvert + 1)$	$\alpha = 2$	0.95679	0.87805	0.89527	0.67372	0.88827	0.69832
$h(x) = \sqrt{\lvert x\rvert}$	$\alpha = 2$	0.95331	0.86760	0.88419	0.67170	0.88082	0.67412
$h(x) = \exp(\lvert x\rvert)$	$\alpha = 2$	0.93798	0.84808	**0.87513**	0.65559	0.88268	0.67784
$h(x) = \cosh(x)$	$\alpha = 2$	0.93868	0.84739	0.87613	0.65257	0.88082	0.67039
$h(x) = \text{sigmoid}(\lvert x\rvert)$	$\alpha = 2$	0.93798	0.84739	**0.87513**	0.65458	0.88082	0.67225
# (TP with #tokens ≥ 10)		1435		993		537	

Table 2.2: Results of Experiment 2. Higher is better in a column.

Dataset		*Computer&Internet*		*Health*		*Society&Culture*	
ACC@#(tokens deleted)		1	5	1	5	1	5
SA		0.01287	0.06436	0.00436	0.02964	0.01174	0.03772
$h(x) = x$	$\alpha = 1$	0.00396	0.04257	0.00349	0.01569	0.00168	0.03437
$h(x) = \tanh(x)$	$\alpha = 1$	0.00396	0.04158	0.00349	0.01569	0.00168	0.03437
$h(x) = x^2$	$\alpha = 1$	0.00594	0.05743	0.00436	0.02528	0.00754	0.04023
$h(x) = \log(\lvert x\rvert + 1)$	$\alpha = 1$	0.00396	0.04257	0.00349	0.01569	0.00168	0.03437
$h(x) = \sqrt{\lvert x\rvert}$	$\alpha = 1$	0.00198	0.03465	0.00436	0.01569	0.00168	0.04107
$h(x) = \exp(\lvert x\rvert)$	$\alpha = 1$	0.00099	0.03861	0.00349	0.01831	0.00419	0.04778
$h(x) = \cosh(x)$	$\alpha = 1$	0.00099	0.03762	0.00349	0.01918	0.00419	0.04862
$h(x) = \text{sigmoid}(\lvert x\rvert)$	$\alpha = 1$	0.00099	0.03861	0.00349	0.01831	0.00419	0.04778
$h(x) = x$	$\alpha = 2$	0.05347	0.05545	0.09939	**0.10462**	0.06035	0.08550
$h(x) = \tanh(x)$	$\alpha = 2$	0.05347	0.05644	0.09939	**0.10462**	0.06035	0.08550
$h(x) = x^2$	$\alpha = 2$	0.03663	0.03861	0.07498	0.09416	0.05616	0.09137
$h(x) = \log(\lvert x\rvert + 1)$	$\alpha = 2$	0.05347	0.05644	0.09852	0.10462	0.06035	0.08550
$h(x) = \sqrt{\lvert x\rvert}$	$\alpha = 2$	**0.05743**	0.05842	**0.10201**	0.09154	**0.06119**	**0.09053**
$h(x) = \exp(\lvert x\rvert)$	$\alpha = 2$	0.05446	0.07030	0.07149	0.08195	0.04946	0.07712
$h(x) = \cosh(x)$	$\alpha = 2$	0.05545	**0.07129**	0.07149	0.08282	0.04946	0.07460
$h(x) = \text{sigmoid}(\lvert x\rvert)$	$\alpha = 2$	0.05446	**0.07129**	0.07149	0.08282	0.04946	0.07628
# (FP with #tokens ≥ 10)		1010		1147		1193	

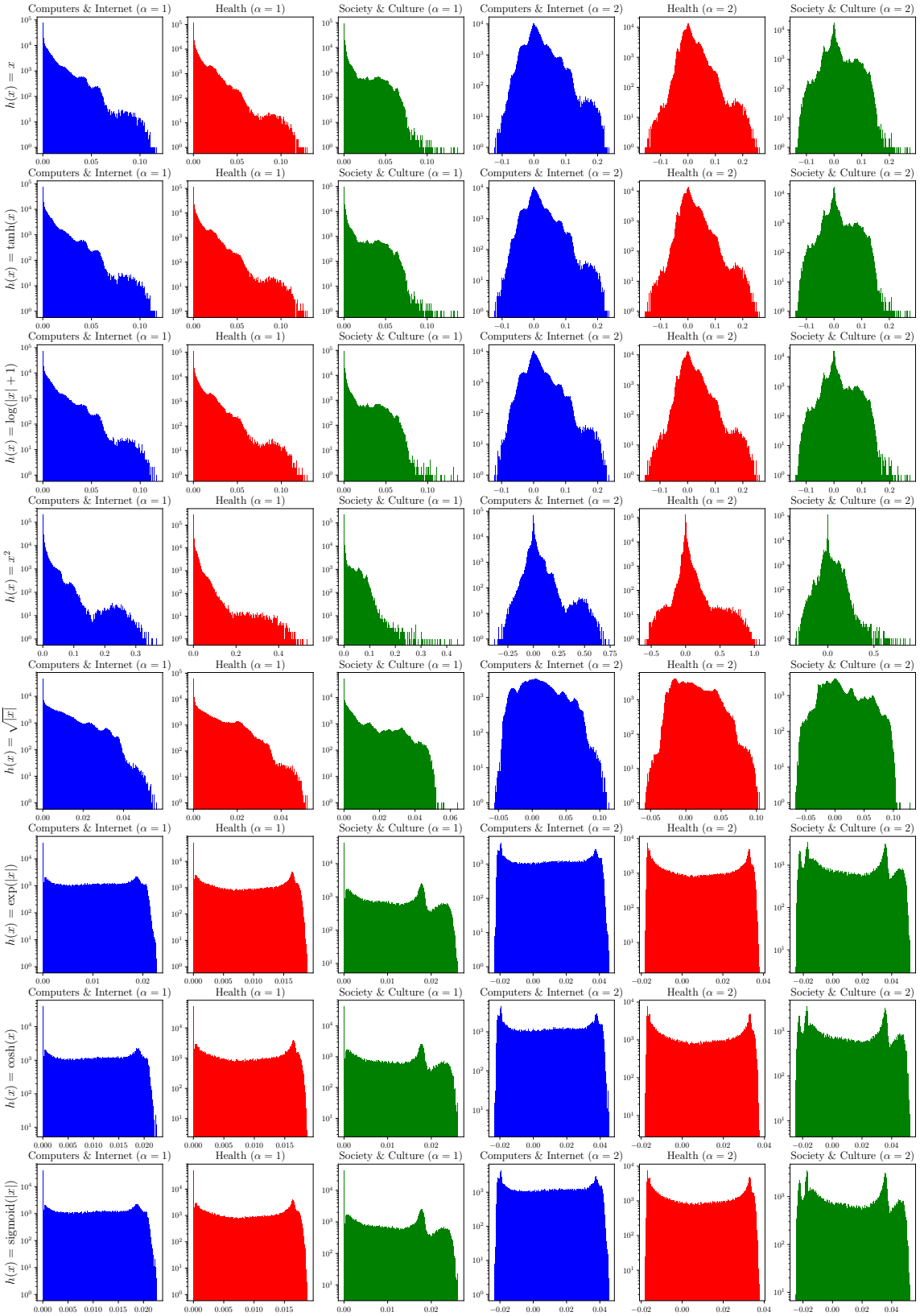


Figure 2-2: Value distributions of the coefficients to divide the relevance scores from pooling-weighted layers by different functions. The number of bins in each histogram is 400.

2.2.5 Example of Visualization

By using the visualization of MI, we can understand the model prediction intuitively. Figure 2-3 shows an example of the visualization on an FP sample in Computer & Internet. We found that introducing nonlinear functions made the relevance scores of inputs sharper and more distinct from others, helping people capture keywords more easily and quickly. In the example, the visualization results by NRP (*i.e.*, LRP_SQRTABS_AS_PL and LRP_EXPABS_AS_PL) represented the keyword "upgrade" with more positive importance than the linear setting (LRP_LINEAR_A2).

Figure 2-3: Examples of the visualization. Red or blue tokens denote positive or negative elements for the predictive result. The deeper color is, the higher relevance is.

2.3 Key Input Subset Sampling

Using model-specific approaches may need to design specific rules for the network architectures. For the sake of adapting ambiguous models, model-agnostic methods are the option. KISS can be a model-agnostic forward approach or a hybrid method by collaborating with backward information [52].

We assume that more essential elements for the model output have higher absolute values of explanatory scores in MI, that is to say, there could be a subset $S \subseteq [n] = \{1, \ldots, n\}$ within which the input elements $\boldsymbol{x}_S$ contribute to the predictive output more significantly than the others. We call those input elements with indices in S *key input elements* (KIEs) in this work. Because it costs $O(2^n)$ operations to select candidate input subset, we can utilize sampling to reduce the computational complexity and estimate the explanatory scores.

2.3.1 Theoretical Basis: Energy-based Model

KISS is based on the energy-based model (EBM) theory, which gives a physical explanation to model training [53]. Given data X and the answers $\mathcal{Y} \ni Y$, the objective is that the energy surface of the model needs to be pushed down to the correct answer Y and pulled up from the other answers as

$$Y^* = \arg\min_{Y \in \mathcal{Y}} E(Y, X), \tag{2.13}$$

where E is the energy function.

Many loss functions can be adapted in the EBM framework, such as mean squared errors (MSE) for regression, negative log-likelihood (NLL) loss for classification, and margin losses. Since most deep models are trained with these losses, a trained deep model is considered to have already or almost satisfied the EBM objective.

Though the EBM theory can be used in various tasks, we present our method with a classification setting in the following sections.

2.3.2 Work Assumption and Importance Score

Given a classification model f and an input $\boldsymbol{x}$, the predictive probability $p(c|\boldsymbol{x})$ of the target class $c \in C$ by the trained model is given by the softmax function:

$$p(c|\boldsymbol{x}) = \frac{\exp\left(-E(c, \boldsymbol{x})\right)}{Z} = \frac{\exp\left(E'(c, \boldsymbol{x})\right)}{Z} \tag{2.14}$$

$$Z = \sum_{y \in C} \exp\left(E'(y, \boldsymbol{x})\right), \tag{2.15}$$

where the negative energy output $E' = -E$ can be obtained from the model's inference. Besides, from the NLL loss function $-\log p(c|\boldsymbol{x}) = -E'(c, \boldsymbol{x}) + \log Z$, we can obtain the *Helmholtz free energy* as

$$\mathcal{F}\left(\boldsymbol{x}\right) = -\log Z, \tag{2.16}$$

which helps us measure the useful work obtainable from a closed system [54].

In KISS, we assume that the system's energy reached $E(c, \boldsymbol{x})$ after an input element x_i did work toward the prediction of c. To give an importance score to x_i with respect to c, we can hinder x_i from delivering its information into the prediction to achieve the energy $E(c, \boldsymbol{x}_{\setminus i})$, where $\boldsymbol{x}_{\setminus i} = \boldsymbol{x} \setminus \{x_i\}$ (*e.g.*, set x_i to 0). If the contribution of x_i is independent of the others, we can calculate the difference between $E(c, \boldsymbol{x}_{\setminus i})$ and $E(c, \boldsymbol{x})$ as the contribution. We define the difference as work by referring to physics, given by

$$w_i(c, \boldsymbol{x}) = -\Delta E = \Delta E' = E'(c, \boldsymbol{x}) - E'(c, \boldsymbol{x}_{\setminus i}). \tag{2.17}$$

Examples of the work assumption are shown in Figure 2-4. Due to $w_i(c_1, \boldsymbol{x}) = -\Delta E = 0$, *i.e.*, the energy with $\boldsymbol{x}_{\setminus i}$ equals the one with $\boldsymbol{x}$, x_i has no contribution to the prediction of the class c_1. Likewise, x_i had done positive work for c_2 by $w_i(c_2, \boldsymbol{x}) > 0$, while x_i has performed negative work toward c_1 by $w_i(c_3, \boldsymbol{x}) = 0$. Since the system's temperature is a constant ($T = 1$) by the softmax function, we obtain the inequality $\Delta\mathcal{F} \leq -\mathcal{W}$ by the maximum work principle, where $\mathcal{W}$ is the work done on the

surroundings by the system. We also assume that the system does no work on the surroundings, *i.e.*, $\mathcal{W} = 0$. If computed work w_i make $\Delta\mathcal{F}_i = \mathcal{F}(\boldsymbol{x}) - \mathcal{F}(\boldsymbol{x}_{\setminus i}) > 0$, we reject the contribution in a possibly incorrect assumption by setting w_i to 0.

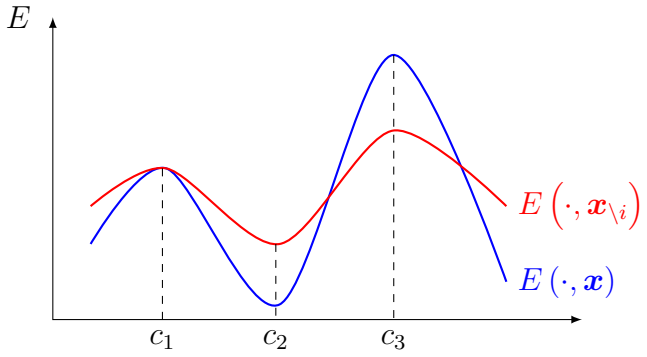


Figure 2-4: Energy surface of a model changed by an input element. We assume that the red curve is the prior step and the blue one is the later step.

On the above assumption, we define the importance score based on the EBM theory as follows.

Definition 3 (EBM-based importance score (EBIS))**.** *For an EBM objective satisfying model, the importance s_i is the expectation of the work done by the input element $x_i \in \boldsymbol{x}$ to the class c's prediction as*

$$s_i = \mathbb{E}\left[w_i(c, \boldsymbol{x})\right]. \tag{2.18}$$

2.3.3 Input Subset Sampling

However, due to the complex connections within a deep model, removing one element from a high-dimensional input may lose some correlation with the other elements. Thus, we generalized the contribution from one input element to one for a non-empty input subset S. Then, we can estimate the work of x_i by dividing the subset contribution by the size of S:

$$w_S(c, \boldsymbol{x}) = E'(c, \boldsymbol{x}) - E'(c, \boldsymbol{x}_{\setminus S}) \tag{2.19}$$

$$\hat{w}_i(c, \boldsymbol{x}) = \frac{w_S(c, \boldsymbol{x})}{|S|}, \tag{2.20}$$

where $|S|$ is the size of S, and $\boldsymbol{x}_{\setminus S} = \boldsymbol{x} \setminus \boldsymbol{x}_S$ indicates the input $\boldsymbol{x}$ masking the elements in S. Equation 2.19 is on the assumption that the work of S is independent of the others. Meanwhile, the rejection condition turns to $\Delta\mathcal{F}_S = \mathcal{F}(\boldsymbol{x}) - \mathcal{F}(\boldsymbol{x}_{\setminus S}) > 0$.

To compute the subset contribution, we have another assumption that the probability of occurrence of x_i is under a distribution $p(x_i|\boldsymbol{x}_{\setminus i})$, and the work done by $\boldsymbol{x}_S$ to c is the expectation of the contributions given by an unknown function g:

$$w_S(c, \boldsymbol{x}) = \mathbb{E}_{i \in S, x_i \sim p(x_i|\boldsymbol{x}_{\setminus i})} \left[g(c, x_i) \right]. \tag{2.21}$$

Because it is hard to model $p(x_i|\boldsymbol{x}_{\setminus i})$ and function g, we use the importance sampling technique to estimate the EBIS.

For a high-dimensional input such as embeddings of words or pixels of an image, globally drawing subsets may have a high chance of obtaining combinations with less interdependence. Therefore, we sample elements in a sliding window style and expect to capture subsets with a higher correlation. Let W_{ij} be the j-th window from which m i.i.d. subsets $S \ni i$ are sampled. By the importance sampling technique, we can estimate the EBIS of x_i to c by

$$\begin{aligned}
\bar{s}_i &= \frac{1}{T_i} \sum_{W_{ij}} \hat{s}_{ij} && (2.22)\\
&= \frac{1}{T_i} \sum_{W_{ij}} \mathbb{E} \left[w_i(c, \boldsymbol{x}) \right] && (2.23)\\
&= \frac{1}{T_i} \sum_{W_{ij}} \frac{1}{m} \sum_{x_i \sim q_j} \frac{p(x_i|\boldsymbol{x}_{\setminus i}) g(c, x_i)}{q_j(x_i)} && (2.24)\\
&= \frac{1}{T_i} \sum_{W_{ij}} \frac{1}{m q_j(x_i)} \sum_{S \ni i} \frac{w_S(c, \boldsymbol{x})}{|S|} && (2.25)\\
&\approx \frac{1}{T_i \times |S|} \sum_{S \ni i} \left(\sum_{W_{ij}} o_{ij}^{-1} \right) \left[E'(c, \boldsymbol{x}) - E'(c, \boldsymbol{x}_{\setminus S}) \right], && (2.26)
\end{aligned}$$

where T_i is the frequency of x_i as the sampling population, q_j denotes the sampling distribution from W_{ij}, and o_{ij} counts the frequency of occurrence of x_i from W_{ij}. Equation 2.23 is obtained by Equation 2.18, and Equation 2.24 is by Equation 2.21 and importance sampling. Then, Equation 2.25 is derived from Equation 2.20. Finally, the

EBIS is approximated by the law of large numbers with Equation 2.19 in Equation 2.26. Counters o_{ij} are used instead of q_j to avoid complicated computation of probability, while T_i scales the estimations as the numbers of windows including x_i may be different from the one of others.

2.3.4 Weight for Sampling and Anisotropic Saliency

In the subset sampling mentioned above, we can naively draw elements from a uniform distribution without replacement. However, there may be a high possibility of getting non-KIEs, leading to underestimation of the importance scores.

By referring to model-specific approaches, we can leverage the gradient information as a sort of weight for the weighted sampling without replacement [55]. For the sake of using gradients, we designed anisotropic saliency (AS) for the sampling weight as follows.

Definition 4 (Anisotropic saliency (AS))**.** *Given an input $\boldsymbol{x} \in \mathcal{X}$, candidate classes C and a model $f : \mathcal{X} \mapsto \mathbb{R}^{|C|}$, the saliency $a_i \in \mathbb{R}$ of the input element $x_i \in \boldsymbol{x}$ is the max norm of gradients over the classes:*

$$a_i = \max_C \left| \frac{\partial f(\boldsymbol{x})}{\partial x_i} \right| + \epsilon, \tag{2.27}$$

where ϵ is a small positive value to avoid zero-weights.

The term *anisotropic* suggests that the weights of elements have different natures due to the gradients derived from different classes. We believe that AS helps us select KIEs with a higher chance by capturing the cross-class backward importance for each input element. For example, given an image of the digit *4* written similarly to a "9", it is expected that the model interpreter to select not only the positive pixels for the class "4" but also negative ones against the class "9".

2.3.5 Algorithm

In this section, we summarize the proposed method in Algorithm 2. The computation for E' at Line 23 is the principal cost by present deep learning architectures. Let $O(|f|)$ be the complexity of the model's forward computation, which can be accelerated using parallel computing and GPUs. Then, the complexity of Algorithm 2 with uniform sampling at Line 12 is $O(m|\mathbb{W}|(|S|\log|S|+|f|))$ or with AS at Line 12 is $O\left(m|\mathbb{W}|\left(|S|\left(1+\log\frac{l^2}{|S|}\right)+|f|\right)\right)$ [56].

2.3.6 Results

In the experiments, we used an altered pixel flipping test (Algorithm 3), which evaluates by the gross accuracy rather than the average normalized predictive output [39], because a change by an input element could not only affect the correct answer but also be even more significant to the other ones. We considered that the gross accuracy could handle all the influences over the candidate answers.

We trained a simple CNN model for MNIST[3] and a ResNetV2 [57] model for CIFAR10[4] on the whole train set with 50,000 samples for each dataset. The whole test set with 10,000 instances are used to perform the pixel flipping test in the evaluation. Table B.2 shows the architecture of the CNN model for MNIST, which was implemented in PyTorch. For the sake of comparison, we employed saliency, Deconvolution (Deconv), GradientShap, Guided Backpropagation (GBP) [58], DeepLIFT, Input $\odot$ Gradient (IXG) [59], Integrated Gradients (IG) [60], and Occlusion-1 [33] as the contrastive methods. Besides, the relevance scores were obtained by masking RGB-channel pixels simultaneous as one input element as $x_i = \left[x_i^R, x_i^G, x_i^B\right]$ in our methods and Occlusion-1, while the ones were the sums of the scores from the three channels in the other methods. Also, AS in ours was given by $a_i = \max\left(a_i^R, a_i^G, a_i^B\right)$. The evaluation results are shown in Figure 2-5, and the areas under the gross accuracy curves (AUCs) by flipping important pixels are shown in Table 2.3.

We observed that our methods outperformed the contrastive approaches on both

[3]http://yann.lecun.com/exdb/mnist/
[4]https://www.cs.toronto.edu/ kriz/cifar.html

Algorithm 2: Key input subset sampling

Input: input $\boldsymbol{x}_{[n]} \in \mathcal{X}$ with n elements, candidate classes C predicated by the trained model $f : \mathcal{X} \mapsto \mathbb{R}^{|C|}$, negative energy function $E' : C, \mathcal{X} \mapsto \mathbb{R}$ of f, size of window $l \times l$, number of elements in a non-empty subset $|S|$, number of i.i.d. samples m.

Output: EBIS's for $\boldsymbol{x}$ with respect to C.

create windows $\mathbb{W}$ over $\boldsymbol{x}$ by the window size $l \times l$ and the stride size 1

$\boldsymbol{a}_{[n]} \leftarrow \mathbf{1}$ for uniform sampling or $\boldsymbol{a}_{[n]} \leftarrow$ AS of $\boldsymbol{x}$ by Equation 2.27 for weighted sampling

$\mathbb{Q} \leftarrow \varnothing$

$[o_{ij}] \leftarrow \mathbf{0}$, a counter matrix where j is the order of the window having i-th input element, $1 \leq i \leq n$, $1 \leq j \leq |\mathbb{W}|$

$\boldsymbol{T}_{[n]} \leftarrow$ frequencies of elements occurring in $\mathbb{W}$

$\boldsymbol{b}_{[n]} \leftarrow \mathbf{0}$

$\mathbb{S} \leftarrow \varnothing$

$\boldsymbol{s}_{[n \times |C|]} \leftarrow \mathbf{0}$

foreach *window* $W \in \mathbb{W}$ **do**

  for $k \leftarrow 1$ **to** m **do**

    $\boldsymbol{a}_W \leftarrow$ sampling weights of the elements in W

    $S \leftarrow$ subset of indices of the elements sampled from W without replacement with the weight $\boldsymbol{a}_W$

    ENQUEUE($\mathbb{Q}$, S)

while $\mathbb{Q} \neq \varnothing$ **do**

  $S \leftarrow$ DEQUEUE($\mathbb{Q}$)

  foreach *index* $i \in S$ **do**

    $o_{ij} \leftarrow o_{ij} + 1$, where $S \subset W_{ij}$

  $\mathbb{S} \leftarrow \mathbb{S} \bigcup S$

forall o_{ij} **do**

  $b_i \leftarrow \frac{1}{T_i} \sum_{W_{ij}} o_{ij}^{-1}$

foreach *subset* $S \in \mathbb{S}$ **do**

  $\boldsymbol{x}_{\setminus S} \leftarrow$ a vector comprehension $[i \in S \ ? \ 0 : \text{copy}(x_i)$ **for** $i \leftarrow 1$ to $n]$

  $\boldsymbol{w}_S \leftarrow \left[\mathcal{F}(\boldsymbol{x}) - \mathcal{F}(\boldsymbol{x}_{\setminus S}) > 0 \ ? \ 0 : \left(E'(c, \boldsymbol{x}) - E'(c, \boldsymbol{x}_{\setminus S})\right)\right.$ **for** $c \leftarrow 1$ **to** $|C|]$ with Equation 2.15 and 2.16

  foreach *index* $i \in S$ **do**

    $\boldsymbol{s}_i \leftarrow \boldsymbol{s}_i + b_i \boldsymbol{w}_S$

return $\boldsymbol{s}/|S|$

Algorithm 3: Pixel flipping test

Input: images $X \ni \boldsymbol{x}$, $\boldsymbol{x} = [x_i]$, the maximum number of token deleted K, a trained model f, relevance $\boldsymbol{r} = [r_j]$ in descending order.

foreach $\boldsymbol{x} \in X$ **do**
  sort the pixels $[x_i]$ into $\left[x'_j\right]$ by the order of $\boldsymbol{r}$
  for $j \leftarrow 1$ **to** K **do**
    $x_i \leftarrow -x_i$ where $x_i = x'_j$
    record the model output $f(\boldsymbol{x})$
obtain the gross accuracy reduction curves for each pixel flipping

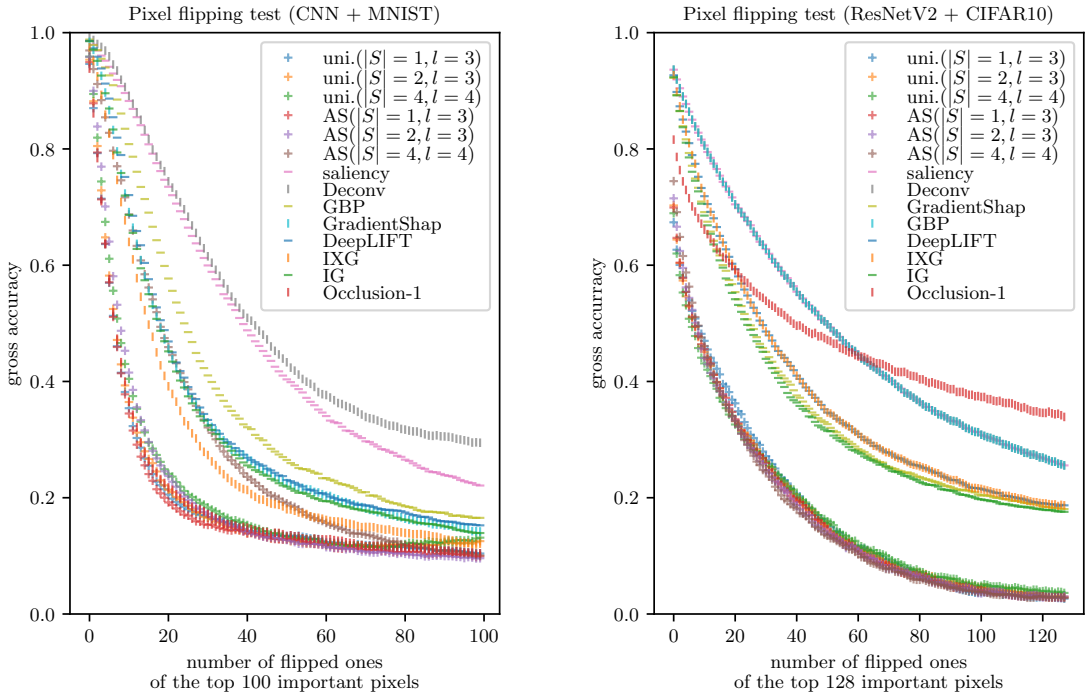


Figure 2-5: Pixel flipping tests on MNIST ($m = 20$) and CIFAR10 ($m = 10$).

tasks. As the handwritten digit images in MNIST are in simple grayscale, the methods only masking one input element each time, such as KISS with $|S| = 1$ or Occlusion-1, had better performance. However, KISS by $|S| = 4$ with AS could overestimate for these simple data. For the CIFAR10 task, it is encouraging that ours' gross accuracy curves were far more convex than the others. We considered two possible reasons for that: 1) the other methods only focused on the target classes, but KISS overlooked all candidate classes by considering the free energy, and 2) the relevance scores computed by the model-specific methods contained different signs and values along the RGB channels, while KISS treated the RGB channels as one element to avoid counteracting contributions by using the forward style.

Moreover, the AUCs of saliency, Deconvolution, and Guided Backpropagation obtained the same results on the ResNetV2 model because of the pre-activation architecture and the residual units without biases. The results of DeepLIFT and Input $\odot$ Gradient showed a similar phenomenon. Besides, we found that AS helped improve the performance of sampling from the AUC results.

Table 2.3: AUCs by flipping pixels. The smaller is the better since it implies that the order by relevance is more meaningful than the others.

Method	**CNN+MNIST@100**	**ResNetV2+CIFAR10@128**
saliency	48.9028	61.7073
Deconv	52.1258	61.7073
GradientShap	32.7096	45.0386
GBP	37.9767	61.7073
DeepLIFT	33.5445	47.7111
IXG	28.5271	47.7111
IG	32.1853	44.047
Occlusion-1	19.973	60.1107
AS ($\|S\| = 1, l = 3$)	**18.9298**	21.4046
AS ($\|S\| = 2, l = 3$)	20.332	21.14
AS ($\|S\| = 4, l = 4$)	29.7233	**21.0196**
uni.($\|S\| = 1, l = 3$)	19.6733	22.1864
uni.($\|S\| = 2, l = 3$)	20.0143	21.8488
uni.($\|S\| = 4, l = 4$)	21.1498	22.0663

Despite remarkable results in the evaluation, the selection for the best parameters

is still an open problem. A possible approach is as like as the one for NRP. We can launch the tests on a small data subset for the selection by the higher performance. In the preliminary experiments, we had tried a set of parameters for pixel flipping tests on a subset with 1000 instances, where $|S| \in \{1, 2, 3, 4, 8\}$, $l \in \{2, 3, 4, 8\}$ and $|S| \leq l$. Then the combinations were chosen for the experiments since they had relatively low AUCs. Another limitation of KISS is that the computational complexity is higher than the one of a model-specific method, usually $O(|f|)$. Although the inference can be accelerated with GPUs and parallel computing, more studies on the sampling approach still need to be done to reduce complexity, such as pruning by weight.

2.3.7 Examples of Visualization

The following two figures are two examples of visualization on the importance scores by KISS. The red or blue pixels denote the corresponding elements' positive or negative contributions with respect to the labels. Additionally, The brightness of each contribution is directly proportional to the normalized EBIS over the classes.

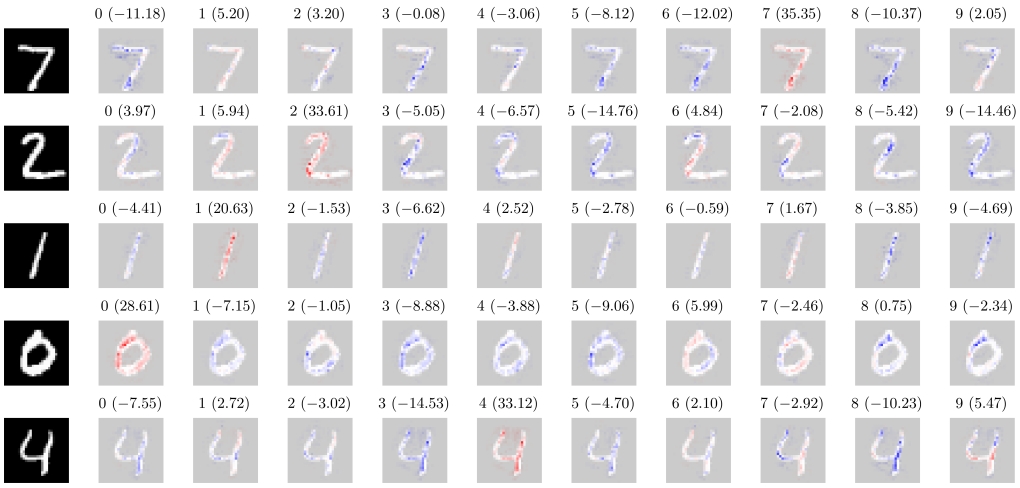


Figure 2-6: Visualization on MNIST with a CNN model ($|S| = 1$, $l = 3$, $m = 20$ with AS). The leftmost images are the input images, while the titles on top of the importance plots are the corresponding labels along with the negative energies E'. The classes with the largest E' over the same rows have the highest predictive probabilities.

In the visualization on the MNIST model (Figure 2-6), KISS could present the following three types of explanations:

- **Support**. All the correct predictions show strong red pixels on the strokes and even the black backgrounds as features that could be highly related to the corresponding classes. However, some strokes can feature in incorrect classes, which denotes these pixels could be similar to those labels, *e.g.*, the written *4* to the class "9" in the bottom row.

- **Protest**. Strong blue pixels were attached to possibly incorrect strokes as protests for the corresponding classes, such as the handwritten *7* to the class "0" in the first row.

- **Shortage**. Sometimes, there are some blue points given by KISS on the backgrounds as supplements which could be the missing features for the regarding labels, *e.g.*, the written *7* to the class "8" in the first row.

Figure 2-7: Visualization on ImageNet with the MobileNetV2 model ($|S| = 10$, $l = 5$, $m = 5$ with AS). Both input images were labeled as Siberian husky. The model predicted the upper image as a Siberian husky yet the lower one as an Eskimo dog.

The visualization on the ImageNet dataset was generated by using KISS on a pretrained MobileNetV2 [61] model (Figure 2-7). It is known that Eskimo dogs have white hair and live in snowfields usually; most of the related images in the training set could satisfy this impression. For the upper image, the importance scores showed that the black fur could be against Eskimo dog. Furthermore, for the lower input

getting incorrect prediction by the model, KISS pointed out that the white snow on the ground could cause the failure.

2.4 Survey and Discussion

We conducted a questionnaire survey to investigate the effect of MI's helping knowledge discovery on images. We used the visualization results by KISS on six samples from MNIST with simple CNN and ImageNet with MobileNetV2.

In Questionnaire (A) (Figure A-1 and A-2), we designed the questions to obtain respondents' opinions on the images without any information by deep learning. In Questionnaire (B) (Figure A-3 and A-4), we introduced the visualization results by KISS and asked respondents whether they agree with the interpretations and discover novelty.

We received 31 responses replied in English, Japanese, or Chinese. Since the questions were non-factoid, we summarized the answer patterns in Table B.3, and human explanation patterns in Table B.4 and B.5.

According to the answer patterns, we found that people tend to hold on to their first judgments, as most of the respondents did not change their answers despite different predictions given by the models. In the following, we discuss our discovery from the respondents' explanations with representative responses for each question.

In Question (1), the main difference in opinion was on the upper part's shape. Most of the respondents answering "4" mentioned the sharp corner on the left, which was also annotated with positive pixels by KISS. A reply said, "I agree with the computer because it says that the character is like both '4' and '9' but more similar to '4' considering the left end of the character." On the other hand, a dissenter found negative annotations on '9' and claimed, "Not really. It looks like the computer considers the angel as a more important factor than the others. However, more factors make it more like a 9 to me." In both the two comments, the respondents perceived the left angel annotated by KISS.

In Question (2), almost all of the respondents answered "9". However, the reasons

for their final judgments were various since there were ten human explanation patterns for the disagreement. A respondent answered "4" and noticed the model's judgment on the upper hole, "No. But I understand the computer's choice because of the straight line here in '9'. I think machine learning is good at noticing negative evidence." Another respondent also indicated the hole but answered "9", "Disagree. The red circle is not closed, this feature can often be observed in the handwritten 9" Obviously, they both mentioned the right upper part of the digit where KISS gave different contribution scores to "4" and "9".

People may have different minds for the handwritten way in Question (3) based on their cultural backgrounds. Most of the respondents who answered "7" treated the middle line as the dash to avoid confusion with other Latin characters. Some respondents who answered "2" noted the last returned stroke. A respondent agreed with the model's explanation on the strong supports in the middle and found the novelty, "Yes. In this case, it is interesting to find the cross-point of the two lines came to be the evidence. This differs from human's cognition of '7'." Another respondent thought it is a "7" by intuition at first, "The first sight will give me the only answer – 7, and even I want to consider another option, I cannot come up with any." After observing the interpretation, he/she changed the answer, "I realised it can also be a 2. And I think the horizontal line really indicates it is the last stroke of the 2." Here, the interpretation helped them discover novelty in the writing way of the strokes.

In Question (4), some respondents who answered "bottlecap" considered the object to be convex or judged by the text on it. Nevertheless, some others thought it to be concave by the shadow or focused on the rim's shape. One agreed with the interpretation on "bottlecap" and wrote, "It judged on the red part and the rim as the same as I did." However, another altered his/her mind from "bottlecap" to "tray" and questioned the model's explanation on the rim, "No. I have changed my opinion. Not like a bottlecap because there is no screw thread and jag on the rim. I disagree with the blue parts for 'tray' and don't know what's the meaning." The different annotations on the rims of "bottlecap" and "tray" led to individuals' divergent explanations.

The object in the image of Question (5) should be an opened traffic light lying

down on the ground. It was a complex context for the CNN-based model without prior knowledge. In the responses, people mainly focused on the characteristic three colors of the object. A respondent thought the model had no idea of the meaning of the colors, "Traffic lights. The red part helps me to find the details of the picture. But I don't agree with the computer's answer. I think maybe the computer didn't recognise the color. Green yellow and red, it should be traffic lights." Another respondent pointed out the disadvantage of the CNN model, "Disagree. Though it looks like face powders, the computer could recognize this is a traffic light if it has learned an opened traffic light – the computer lacks prior knowledge." They both found the defects of the model based on the visualization.

In Question (6), almost all of the respondents answered "beer bottle" based on the text or the brand. A respondent considered the model illiterate because there was no annotation on the text, "I don't agree. The result tells us the computer can only recognise the shape. It can't recognise the words, no matter English or Chinese." Another respondent was focusing on the surface curve and indicated the negative annotations on "beer bottle", "Yes. The color explains my reason for the choice! The curve of the surface is not an exact surface of a beer bottle." Here, the model's explanation help people reflect on the model or their consideration.

According to the above discussion, we observed that the interpretation by KISS helped people notice blind spots, reasons of the intuition, or defects of the models since the respondents reviewed their knowledge and used pointers or words to explain their discovery based on the interpretation results. Therefore, we can expect MI approaches to facilitate the understanding of deep models' prediction, the consideration of deep models' reliability and improvement, and novelty discovery. Moreover, it is interesting that more than half of the respondents replied and persisted the answers different from the labels in Question (3) and (4). Also, the likenesses of controversial answers given by the models were very close in Question (2), (3), and (6). This evidence suggests that we should adopt deep models more flexibly and carefully, such as allowing the models to warn human users of near predictive results for reviews.

2.5 Related Work

As the development of studies on model interpretation in deep learning, various methods for explaining deep models have been proposed.

2.5.1 Model-agnostic methods

LIME trains explainable models (*i.e.*, LASSO and decision trees) to predict local spaces by sampling data around an input [30]. KernelSHAP combines LIME and game-based approximation to estimate the importance [31]. In soft decision trees, the prediction is re-fit by using the trees in which the criteria of splits are logistic regressions with temperature [32]. DeepVis is an element subset sampling method similar to KISS, but it approximates $p(x_i|\boldsymbol{x}_{\setminus i})$ with Gaussian distribution and computes the average weights of evidence (WoE) for each sample as contribution scores [34]. L2X is an information-based method that maximizes the mutual information between an input subset and the output by training a neural network [36]. In SIScollection, minimal subsets whose outputs are greater than a preset threshold are selected as the explanation [35].

2.5.2 Model-specific methods

Saliency maps, such as sensitivity analysis [37], Guided Backpropagation (GBP) [58], Input $\odot$ Gradient (IXG) [59], Integrated Gradients (IG) [60], GradCAM [62], utilize the information of gradients of the output to each input element by chain rule. However, some of them could result in misleading the explanation [38]. DeepLIFT is a relevance decomposition method similar to LRP by using baseline inputs to estimate positive and negative contributions [40]. GradientSHAP estimates the importance by combining game-based approximation and gradients [31].

2.5.3 Other Taxonomy

The above categories of MI methods are by whether the approach is dependent on a specific model. A superclass of MI is Explainable AI (XAI) [12, 63–66].

By scope, XAI methods can be classified into local and global approaches. Local methods generate an explanation for an individual instant, while global methods help understand the whole mechanism of a model [67, 68]. Another taxonomy of XAI in DL is according to the space-time relation of a method. Pre-model methods are used before the model inferences to analyze properties of samples, *e.g.*, principal component analysis (PCA) [69, 70] and t-distributed stochastic neighbor embedding (t-SNE) [71]. Some network architectures can extract interrelation in a given input, such as the attention mechanism [72, 73], treated as in-model methods. Naturally, the interpretation occurs after the model training and prediction is of post-model. Therefore, NRP and KISS are also local post-model interpretation methods.

Chapter 3

Case Study: Deep Learning on Laser Machining

Laser machining, or laser processing, uses an optical process to remove material via the melting or ablation phenomenon between the material and a laser beam [74]. Due to the advantages of laser machining, such as precision, flexibility, automation, and versatility [75], it has been widely used for high-precision processing recently. It is believed that laser machining is a crucial CPS application in Industry 4.0 and Society 5.0 [76, 77].

The case study is on a project in the MEXT Q-Leap program [78]. This project aims at monitoring processing status during laser machining. For the monitoring, Tani *et al.* [20] proposed a low-cost, high-speed data acquisition method in which Fourier speckle patterns of scatterings from the surface of processed material are captured as the image data. They also tried using a ResNet model to predict types of material and volume of ablation. However, This sort of data utilization is still less studied in physics and data science, and it is expected to be applied in more physical tasks. Through the negotiation in an IMDJ workshop, we obtained an opportunity to analyze the dataset and verify DL's performance on it [79].

This chapter will describe the laser machining dataset with exploratory analysis, evaluate the performance of feature extraction with different deep models in single-task learning (STL) or multi-task learning (MTL), and finally introduce applications of

proposed methods on the dataset.

3.1 Analysis on Laser Machining Data

To employ machining learning methods, we needed to preprocess the dataset and do some preliminary analysis to understand it. This section will report on applying principal component analysis (PCA) to the dataset.

3.1.1 Details of The Dataset

The dataset adopted in this work was acquired from laser machining experiments on Silicon materials in 10 different laser power settings. In each power setting, 105 independent experiments were operated, and 250 sequential stages were captured by the camera every 200ms in each experiment. Thus, the dataset included a total of 262,500 images. Each image records Fourier transformed scatterings within 400×4080 pixels in grayscale, and all the images have already been labeled with corresponding powers and shot numbers. For the sake of experimental reproduction, we split the dataset into three subsets by the experiment IDs shown in Table B.6.

Since the raw images' size is too large for our device, we resized them into 224×224 with bilinear interpolation. Existing CNN models mostly use this size. In the preprocessing, each pixel value was normalized by min-max scaling and z-scores with the mean 0.109251 and the standard deviation 0.033309, which were the empirical values over the training set.

3.1.2 Exploratory Analysis

To understand the laser machine data, we decided to apply principal component analysis (PCA) [69, 70] on the training set. Due to the large scale of the dataset with the number of samples $N = 175,000$ and the size of each image $M = 224 \times 224 = 50,176$, we employed singular value decomposition (SVD) alternatively [80] to calculate the sorted singular values $\sigma_{[M]}(\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_M)$ and obtained ordered eigenvalues

$\lambda_{[M]}(\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M)$ by the following equation:

$$\lambda_i = \frac{\sigma_i^2}{N-1}. \tag{3.1}$$

Then, for the evaluation of components explaining variance, we computed the cumulative explained variance ratios (CEVRs) given by

$$R_i = \frac{\sum_{j=1}^{i} \lambda_j}{\sum_{k=1}^{M} \lambda_k}, \tag{3.2}$$

and plotted the results as shown in Figure 3-1.

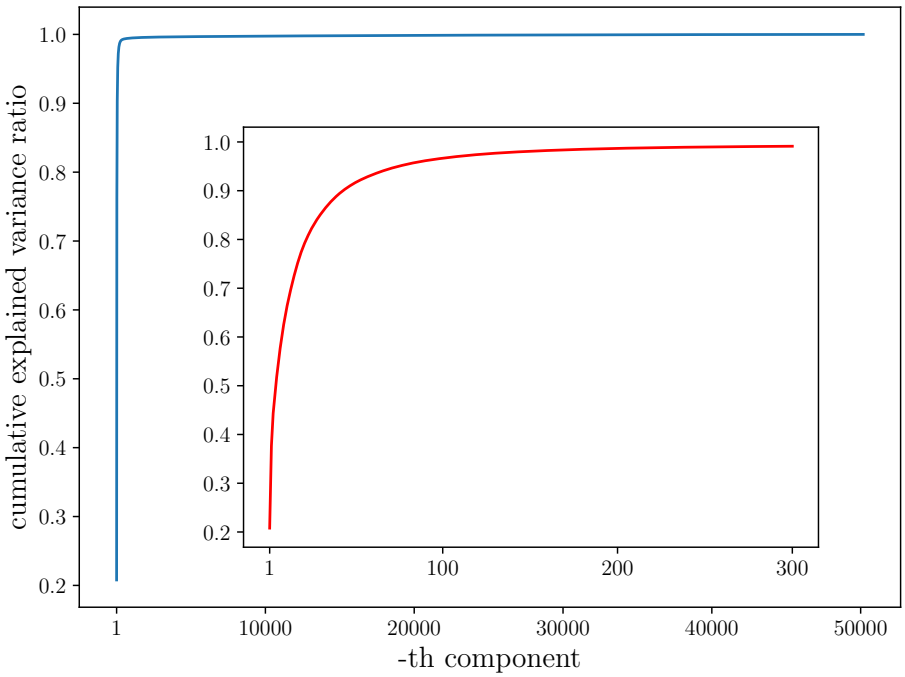


Figure 3-1: CEVRs R_i on the laser machining data. The blue line shows for all components in the training set, while the red one is a zoom-in for $1 \leq i \leq 300$.

According to the CEVRs, we could adopt less than 300 components to obtain variance greater than 99% on the training data. This result helped us alleviate the curse of dimensionality in using some machine learning methods on the data.

3.2 Feature Extraction in Multitask Deep Learning

Feature extraction is a critical step for the utilization of deep learning in downstream applications. To evaluate the performance of feature extraction on the speckle pattern data, we designed two tasks as follows:

1. Power Setting Classification (PSC): Given a speckle pattern image, predict one of the ten power settings as the corresponding power that generated the pattern.

2. Shot Number Regression (SNR): Given a speckle pattern image, predict the log of its numerical order in a single experiment.

Since it is a strong constraint that the integral shot numbers are discrete in a specified range $[1, 250]$, the log instead of the discrete integer was for a soft regression in SNR.

Then, we adopted AlexNet [81] and ResNet [82] for different models with the single-task or multi-task objective as follows.

3.2.1 Image Feature Extraction with CNN

As shown in Figure 3-2, we dropped all the fully-connected neural network (FNN) layers at the ends of the two original base models. Then, we connected the last convolutional layers to the concatenated pooling layer. We expect that avg-pooling could transfer the global extracted information of input, while max-pooling could select those significant features.

Generally, ResNet performs better than AlexNet on most image datasets. However, it showed different results on the laser machining data in the later experiments.

3.2.2 Objective Functions

This section is on the loss functions for the two single tasks respectively and a composed objective function for the multi-task.

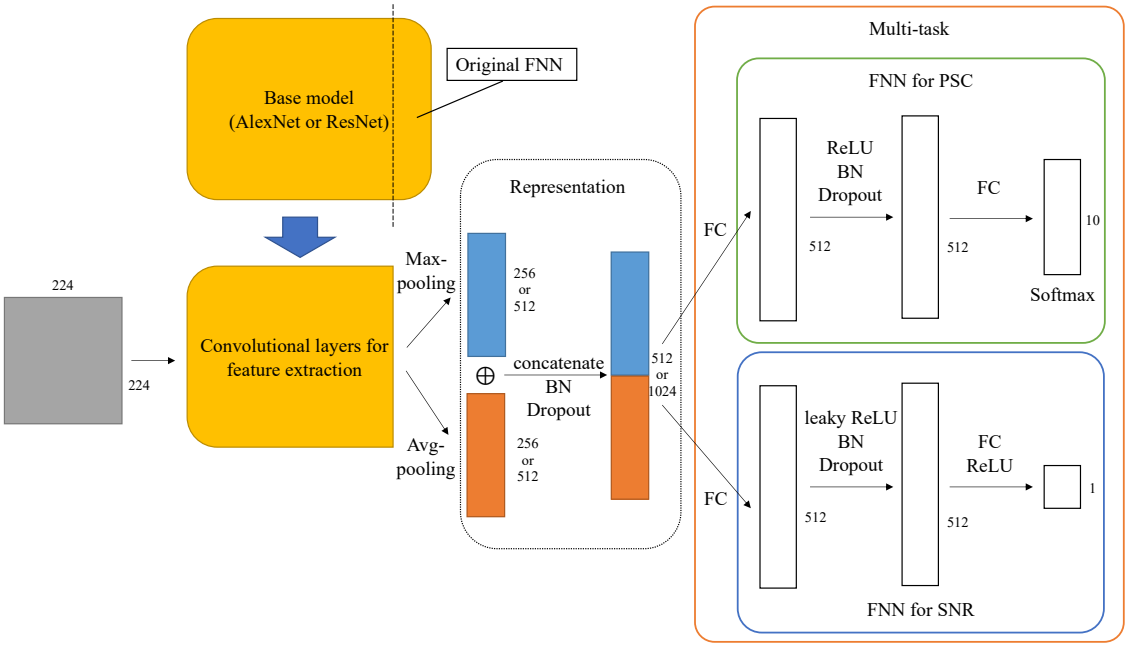


Figure 3-2: Architecture of deep models. The numbers are the sizes of their nearest sides (the default is 1).

For the PSC Task

Since PSC is a typical classification task, we directly utilized the cross-entropy loss [4] defined as

$$\mathcal{L}_{PSC} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{i,k}\log p_{i,k} \tag{3.3}$$

$$y_{i,k} = \begin{cases} 1 & \text{if } c_i = k \\ 0 & \text{otherwise} \end{cases}, \tag{3.4}$$

where for the i-th input sample, c_i is one of the K power setting labels, and $p_{i,k}$ is the predictive probability.

For the SNR Task

For the regression task, we applied smooth L_1 loss [83] as the objective function formulated by

$$\mathcal{L}_{SNR} = \frac{1}{N}\sum_{i=1}^{N} \text{smooth}_{L_1}\left(z_i - z_i'\right) \tag{3.5}$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| \leq 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}, \tag{3.6}$$

where for the i-th input sample, z_i is the log of labeled shot number and z_i' is the predicted logarithm. There are two reasons for using smooth L_1 loss instead of normal L_1 loss or squared error: 1) it could prevent too large gradients being propagated to the upper layers when the absolute loss is greater than 1, and 2) it could do softer learning with smaller gradients when the difference is in the range $(0, 1]$.

For the Multi-task

As sequences of ablation volumes are variant by different laser machining power settings, we believe that the shot numbers are also relative to the power settings. Therefore, we can compose the objectives for PSC and SNR as a multi-task objective.

By sharing the neural network layers of feature extraction to the two different single-task FNNs, we trained the whole model with this global loss function:

$$\mathcal{L} = \mathcal{L}_{PSC} + \mathcal{L}_{SNR} \tag{3.7}$$

3.3 Experiment

In the experiment, we employed accuracy (ACC), precision (PR), recall (RC), and the F^1 score to the evaluation for PSC, while used the mean absolute error (MAE) and the R^2 score for SNR. Then, we trained the mention deep models and baselines to evaluate the performance of feature extraction on the laser machine data.

3.3.1 Metrics

Given N samples and K classes, the metrics for PSC are given by

$$ACC = \frac{1}{N}\sum_{i=1}^{N} 1(c_i = c_i') \tag{3.8}$$

$$PR = \frac{1}{K}\sum_{k=1}^{K} \frac{TP_k}{TP_k + FP_k} = \frac{1}{K}\sum_{k=1}^{K} \frac{\sum_{i=1}^{N} 1(c_i = k \wedge c_i' = k)}{\sum_{i=1}^{N} 1(c_i' = k)} \tag{3.9}$$

$$RC = \frac{1}{K}\sum_{k=1}^{K} \frac{TP_k}{TP_k + FN_k} = \frac{1}{K}\sum_{k=1}^{K} \frac{\sum_{i=1}^{N} 1(c_i = k \wedge c_i' = k)}{\sum_{i=1}^{N} 1(c_i = k)} \tag{3.10}$$

$$F^1 = \frac{1}{K}\sum_{k=1}^{K} \frac{2 \times PR_k \times RC_k}{PR_k + RC_k}, \tag{3.11}$$

where $1(\cdot)$ denotes the indicator function, c_i' is the predictive power setting of i-th sample, and TP_k, FP_k, and FN_k are the numbers of true positives, false positives and false negatives for class k respectively; the metrics for SNR are formulated by

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |z_i - z_i'|, \tag{3.12}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (z_i - z_i')^2}{\sum_{i=1}^{N} \left(z_i - \frac{1}{N}\sum_{i=1}^{N} z_i\right)^2}. \tag{3.13}$$

Except that MAE is the lower the better, the others are the higher the better. Since the numbers of samples in each power settings were the same on the datasets, $ACC = RC$ was obtained.

3.3.2 Model Setups

In the experiment, the PyTorch implementations of AlexNet and ResNet were used as the base models. Behind the deep features layer, a BatchNorm and a 0.25-Dropout were introduced for better stability and generalization. The negative slope of the leaky ReLU was set to 0.3. The loss functions were optimized by SGD with the weight

decay 1.0×10^{-4} and a triangular cyclic scheduler [84]. For each cyclic period of 16.5 epochs, the scheduler adjusted the learning rate in the range $[5.0 \times 10^{-4}, 3.0 \times 10^{-2}]$ and the momentum within $[0.8, 0.9]$. For every epoch, the batch size was 256, and the training images were reshuffled. Every deep model was trained for 100 epochs and the parameters were selected by better ACC or/and MAE and lower loss on the validation set.

As baselines, support vector machine (SVM) [85] and simple FNN were used in the comparison. The linear kernel was applied to the SVM models in this experiment since it was better than the radial basis function (RBF) on the laser machining dataset. According to the result in Section 3.1.2, the image inputs to SVM models were transformed to 260-dimensional data by using truncated SVD [86] due to $R_{260} > 0.99$. Moreover, the simple FNN models' architectures were the same as the ones in Figure 3-2, where input images were flattened into 50,176-dimensional vectors.

3.3.3 Results

With the metrics and the model settings mentioned above, we trained the models in STL or MTL and obtained the results as shown in Table 3.1 and 3.2. According to the results, we found that using AlexNet for the feature extraction in MTL was better than the other comparative models for both tasks. Although ResNet has residual blocks and a deeper architecture, its performance was worse than AlexNet in these experiment settings. A possible reason is that the Fourier transform could be considered as a kind of feature extraction whose "parameters" can not be optimized in the training. As a result, the deeper models could be harder to tune the succeeding layers' parameters, and then the overfitting could occur.

To discuss the advantage of MTL, we also made the confusion matrices of the predictive results by the AlexNet-based models, as shown in Figure 3-3. We found that the values were more concentrated on the diagonal in MTL than in STL. MTL helped reduce errors for most classes, especially for the samples captured in the 1.8mW power setting. Though the MTL model performed a little worse than the STL model for the 3.0mW and 3.5mW settings, their errors were still located at the neighborhoods

Table 3.1: Results of different models for PSC.

Model	**Validation**			**Test**		
	$\boldsymbol{ACC}$	$\boldsymbol{PR}$	$\boldsymbol{F^1}$	$\boldsymbol{ACC}$	$\boldsymbol{PR}$	$\boldsymbol{F^1}$
SVM with SVD	0.44333	0.49753	0.43899	0.52022	0.53973	0.49488
simple FNN in STL	0.71637	0.70822	0.71340	0.73178	0.77212	0.72773
simple FNN in MTL	0.71803	0.72773	0.70776	0.75502	0.78540	0.74073
AlexNet in STL	0.87184	0.88112	0.87064	0.88446	0.89578	0.88406
AlexNet in MTL	**0.90069**	**0.90809**	**0.90032**	**0.9061**	**0.91547**	**0.90524**
ResNet in STL	0.87912	0.89091	0.87580	0.89204	0.90394	0.89191
ResNet in MTL	0.85171	0.87135	0.85183	0.88202	0.89715	0.87770

Table 3.2: Results of different models for SNR.

Model	**Validation**		**Test**	
	$\boldsymbol{MAE}$	$\boldsymbol{R^2}$	$\boldsymbol{MAE}$	$\boldsymbol{R^2}$
SVM in SVD	0.83891	–0.39754	0.83301	–0.37363
simple FNN in STL	0.40982	0.70053	0.41074	0.70823
simple FNN in MTL	0.42202	0.68666	0.41330	0.69938
AlexNet in STL	0.35468	0.73977	0.37303	0.71816
AlexNet in MTL	**0.28893**	**0.84342**	**0.29558**	**0.82798**
ResNet in STL	0.35415	0.76356	0.37520	0.76356
ResNet in MTL	0.32888	0.79420	0.34177	0.78346

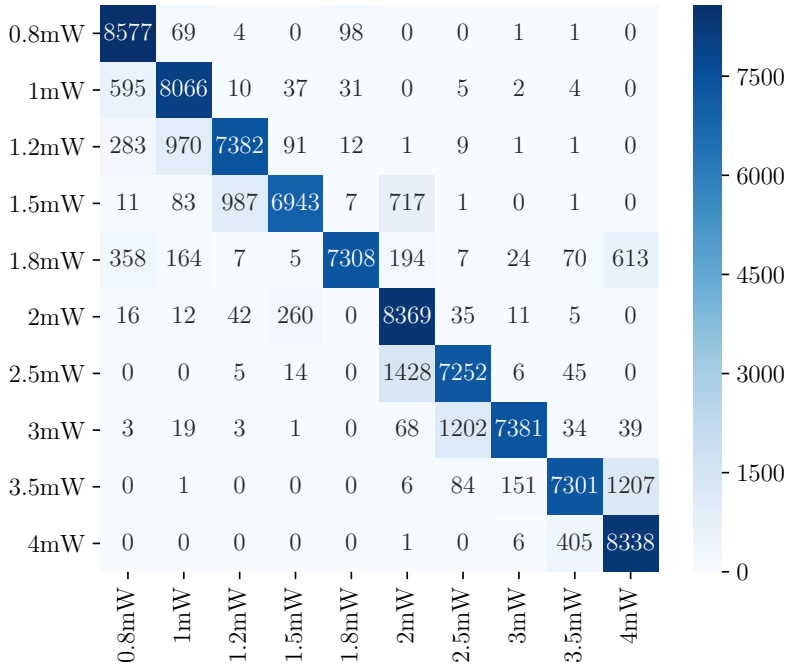
(2.5mW and 4.0mW, respectively).

Also, we plotted ACCs and MAEs over each shot number as shown in Figure 3-4 and 3-5, where the p-values were given by the one-way ANOVA tests. There are obvious turning points near the 25th shots since the ablation volume is too little to carry enough information of scatterings at the beginning of laser machining. However, these plots show that MTL facilitated the predictions also for the beginning of the processing. The reason could be that MTL optimized the model by referring information from the PSC objective and the SNR objective simultaneously, while STL had no more other referable information.

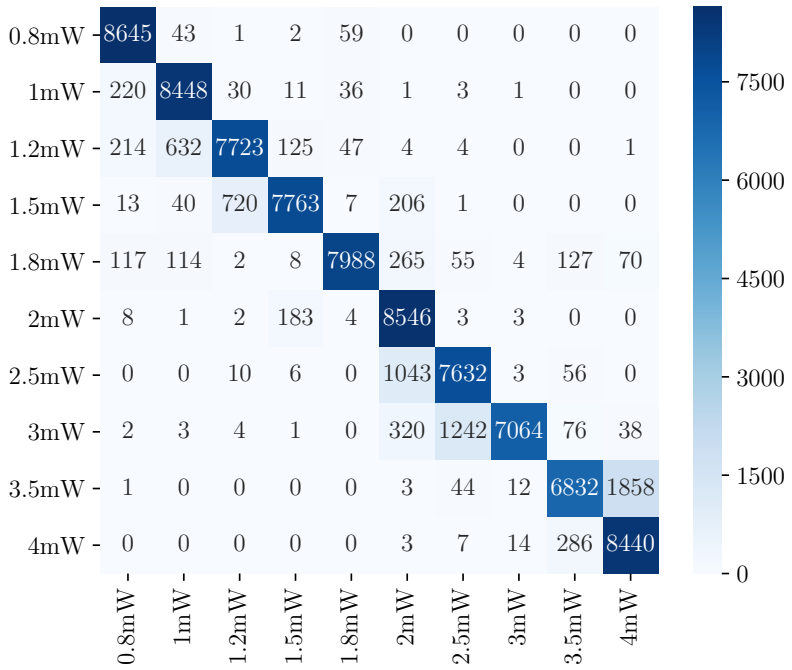
3.4 Discussion

Until now, I presented an evaluation of deep models for the feature extraction of laser machining data, which was motivated by attending the IMDJ workshop and the task of laser processing monitoring. We found that the AlexNet-in-MTL model performed better than the ResNet or STL based model through the experiment. Also, due to the lower computational cost, AlexNet-based models could be more suitable for real-time applications. However, this feature extraction approach was supervised as it depended on the label information (power settings and shot numbers). In the laser machining data, it is considered that some images record anomalies of processing (*i.e.*, outliers or change points), and it is hard to label the anomalies by the recognized information. Thus, the representative features extracted from limited label information could lead to weak generalization for downstream tasks. To solve this problem, unsupervised [87] or self-supervised approaches [88] are the candidates. Additionally, since the target material of the dataset we obtained was only Silicon, we still need more data on other materials for the evaluation.

Also, we attempted applying MI on the AlexNet-based model for the explanation as shown in Figure A-5. We observed that the supports and protests appeared on some speckles or the surrounding area, which is expected to be certain hints for physical experts to elucidate the principles behind laser processing. However, it is still difficult

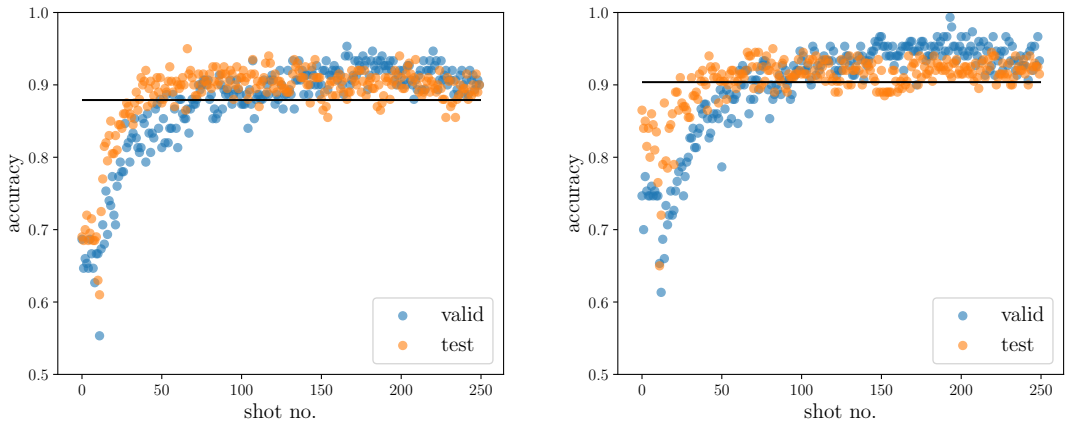


(a) In STL



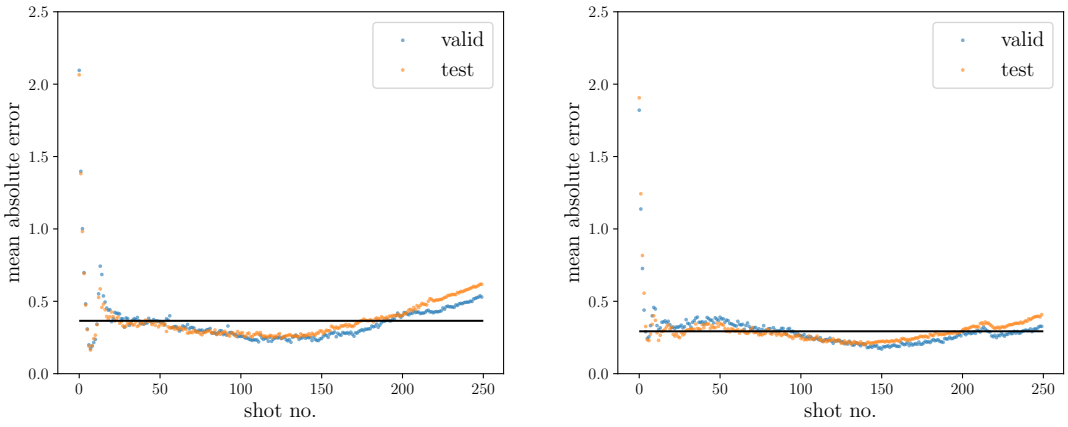
(b) In MTL

Figure 3-3: Confusion matrices of the power setting predictions by AlexNet models in (a) STL and (b) MTL. The power settings on X-axis denote the labels and the ones on Y-axis are the predictive results. The values located at the diagonal are the numbers of corresponding correct predictions, while the others are the numbers of corresponding errors.



(a) In STL (b) In MTL

Figure 3-4: ACC of PSC over each shot number in (a) STL and (b) MTL($p < 0.001$). The black lines denote the ACC over the validation and test sets.



(a) In STL (b) In MTL

Figure 3-5: MAE of SNR over each shot number in (a) STL and (b) MTL ($p < 0.001$). The black lines denote the MAE over the validation and test sets.

to explain the changes among a processing sequence due to KISS is designed for single instances. It is also a motivation behind the method for sequential data described in the next chapter.

3.5 Related Work

The application of DL on laser machining is still a new study field. Mill *et al.* input the images of the laser-machined surfaces to a CNN model to predict material types, laser fluences, and the numbers of pulses simultaneously with a supervised approach [89]. Francis and Bian used stacked thermal images as input and a CNN model to predict the distortion in laser-based additive manufacturing [90]. In the previous work, Tani *et al.* took three continuous speckle pattern photos as one input into a residual neural network (ResNet) to predict ablation depths and the types of materials [20]. In another similar work, McDonnell *et al.* fed spatial images that mirrored Nickle material shaped by three sequential pulses to a deep model for the depth map prediction [91].

Our work focused on investigating the performance of feature extraction with different models in STL and MTL with the information of power settings and the shot orders.

Chapter 4

Sequential Anomaly Detection with Pessimistic Contrastive Learning

Sequential data is a common and important kind of data form in various real-world applications, in which there usually are anomalies, such as outliers and change points. Sometimes, it is hard to label anomalies or extract features from the massive high-dimensional sequential data if there is a lack of prior knowledge, such as the laser machining data described in Chapter 3.

As far as is known, deep learning excels in extracting useful information from high-dimensional data. The anomalies, intended to mark out from raw data, need to be labeled for supervised learning. Obviously, it is a chicken-and-egg problem. Therefore, we consider non-supervised approaches in which it is no need for labels.

This chapter will present pessimistic contrastive learning (PCL), a self-supervised deep learning method for anomaly detection on sequential data. In PCL, we pessimistically assume that there are anomalous data points in the sequences. Then, the anomalies will be recognized by driving the data points to contrast with each other within windows of context.

4.1 Prior Work: SimCLR

The contrastive approach in PCL was inspired by the simple framework for contrastive learning of visual representations (SimCLR) [92], a self-supervised method to extract representative features from images for downstream tasks.

In SimCLR, two different data augmentation functions λ and λ' are sampled from the identical family of augmentations Λ at the beginning. Then, from the same input $\boldsymbol{x}$, the operators λ and λ' generate two correlated instances $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_j$, which are encoded by a neural network f into two representations $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$. Next, $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ are embedded by another network g into two projections $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$. The similarity of $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ is then maximized in the training by minimizing the NT-Xent loss given by

$$\ell(i,j) = -\log \frac{\exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_j\right)/T\right)}{\sum_{k=1}^{2N} \exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_k\right)/T\right)[k \neq i]} \tag{4.1}$$

$$\operatorname{sim}(\boldsymbol{u}, \boldsymbol{v}) = \frac{\boldsymbol{u}^\top \boldsymbol{v}}{\|\boldsymbol{u}\|\|\boldsymbol{v}\|}, \tag{4.2}$$

where T denotes temperature and N is the minibatch size. In Equation 4.1, the lower temperature leads the stronger similarity and vice versa since the term in the logarithm is derived from the Boltzmann distribution as

$$p_i = \frac{\exp\left(-E_i/kT\right)}{\sum_j \exp\left(-E_j/kT\right)}, \tag{4.3}$$

where k is a constant and a term like $\exp\left(-E/kT\right)$ is called the *Boltzmann factor* [54].

If we design the data augmentation appropriately enough, using SimCLR can make neural networks capture the invariant information within two different views of the input for feature extraction without labels.

4.2 Assumptions and Proposed Model

This section will propose a new network model and loss functions to train the model based on the following assumptions.

Given a sequence $[\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_i, \cdots, \boldsymbol{z}_j \cdots, \boldsymbol{z}_{L-1}, \boldsymbol{z}_L]$ $(1 \leq i < j \leq L)$, I model it by using the following pessimistic policy which includes three assumptions for the non-supervised anomaly detection task.

Assumption 1. *The event that an anomaly occurs on the point $\boldsymbol{z}_i$ is independent of the other anomaly occurrences in the sequence.*

Assumption 2. *The event that points z_i and z_j have a relation $r_{ij} \neq 0$ is independent of the other pairs' relationships in the sequence.*

Assumption 3. *If an anomaly occurs on z_i or z_j, the two points lose the relationship, i.e., $r_{ij} = 0$.*

4.2.1 Objective Functions: CARE and SeNT-Xent

Based on the above assumptions, two objective functions will be derived with simple probability and information theories as follows.

Let p_i and p_j be the probabilities of anomalies $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ occurring, and then Assumption 1 implies

$$p_{ij} = p_i + p_j - p_i p_j \tag{4.4}$$

$$\bar{p}_{ij} = 1 - p_{ij}, \tag{4.5}$$

where p_{ij} is the joint probability. According to Assumption 2, the probability of the relationship between $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ is formulated by

$$q_{ij} = \sigma\left(r_{ij}\right) = \frac{\exp\left(r_{ij}\right)}{1 + \exp\left(r_{ij}\right)} \tag{4.6}$$

$$r_{ij} = \frac{\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_j\right)}{\tau\left(i, j\right)}, \tag{4.7}$$

where $\tau\left(\cdot, \cdot\right) > 0$ is the attemperation function that can be constant, linear, or

nonlinear . Moreover, the opposite probability of q_{ij} is

$$\bar{q}_{ij} = 1 - q_{ij} = \frac{1}{1 + \exp(r_{ij})} = \frac{\exp(0)}{\exp(0) + \exp(r_{ij})}. \tag{4.8}$$

Equations 4.3 and 4.8 prove that the formulation of q_{ij} also satisfies Assumption 3 *i.e.*, there is no relation between $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ (*i.e.*, $r_{ij} = 0$) at the opposite event that some anomaly occurred.

To combine the three assumptions above, the loss function to handle the possible anomaly occurrences is defined by

$$\ell_{CARE}(k) = \frac{1}{L-1} \sum_{i=1}^{L-1} \frac{1}{\min(s, L-i)} \sum_{j=i+1}^{\min(i+s,L)} D_\alpha \left(p_{ij}^{(k)} \| \bar{q}_{ij}^{(k)} \right) \tag{4.9}$$

$$D_\alpha(p\|q) = H(p,q) - \alpha H(p) \tag{4.10}$$

$$= -p \log q - \bar{p} \log \bar{q} + \alpha \left(p \log p + \bar{p} \log \bar{p} \right), \tag{4.11}$$

where k is the index of the sample in the minibatch and s is the lookahead size to simulate the sliding windows as shown in Figure 4-1. $D_\alpha(\cdot\|\cdot)$ is the relative entropy altered from Kullback-Leibler divergence by adding an anomaly sampling penalty α along with the entropy of the joint distribution of anomaly occurrence, where the term $\alpha H(p)$ works as a regularization by the principle of maximum entropy. Minimizing the term $D_\alpha \left(p_{ij}^{(k)} \| \bar{q}_{ij}^{(k)} \right)$ makes the distribution of Assumption 3 close to the one of Assumption 1. For convenience, we name this loss function CARE (context-attempered relative entropy loss).

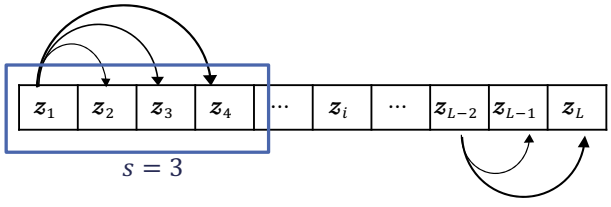


Figure 4-1: Sliding window fashion in CARE. The links denote the relating operation in the training.

For the sequential contrastive learning, Equation 4.1 is extended to SeNT-Xent (sequential NT-Xent) as

$$\ell_{SNTX}(a,b) = -\frac{1}{L}\sum_{i=1}^{L}\log\frac{\exp\left(\mathrm{sim}\left(\boldsymbol{z}_i^{(a)},\boldsymbol{z}_j^{(b)}\right)/T\right)}{\sum_{k=1}^{2N}\exp\left(\mathrm{sim}\left(\boldsymbol{z}_i^{(a)},\boldsymbol{z}_j^{(k)}\right)/T\right)[k\neq a]}. \tag{4.12}$$

Therefore, the global loss function for the training is given by

$$\mathcal{L} = \frac{1}{2N}(\mathcal{L}_{SNTX} + \mathcal{L}_{CARE}) \tag{4.13}$$

$$\mathcal{L}_{SNTX} = \sum_{k=1}^{N}\left[\ell_{SNTX}(2k-1,2k) + \ell_{SNTX}(2k,2k-1)\right] \tag{4.14}$$

$$\mathcal{L}_{CARE} = \sum_{k=1}^{N}\left[\ell_{CARE}(2k-1) + \ell_{CARE}(2k)\right], \tag{4.15}$$

which is a multi-task loss as well. Figure 4-2 sketches the learning framework by minimizing the global loss $\mathcal{L}$, where we can set two different or identical embedding heads for the two tasks.

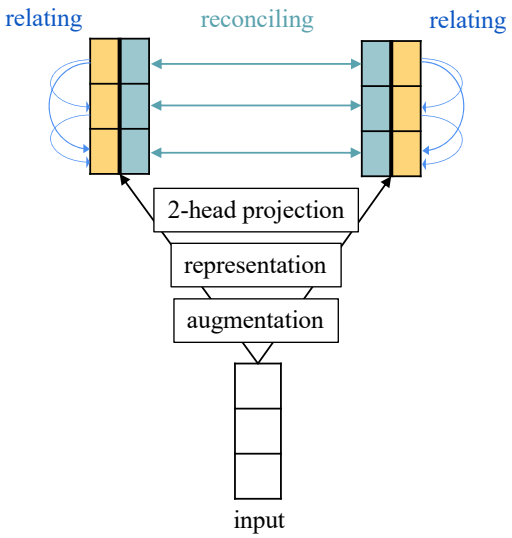


Figure 4-2: Learning framework of PCL. The consistency of each projection pair derived from the same data point is maximized with the loss function $\mathcal{L}_{SNTX}$, while the interrelations among the data points are learned with $\mathcal{L}_{CARE}$.

4.2.2 Network Model: S^{3}ADNet

The three assumptions have been encapsulated into the CARE loss so far, but the probability of anomaly occurrence has not been modeled. Thus, I devised a neural network model called S^{3}ADNet (self-supervised sequential anomaly detection network), as shown in Figure 4-3 to obtain the probabilities at each data points in a sequence.

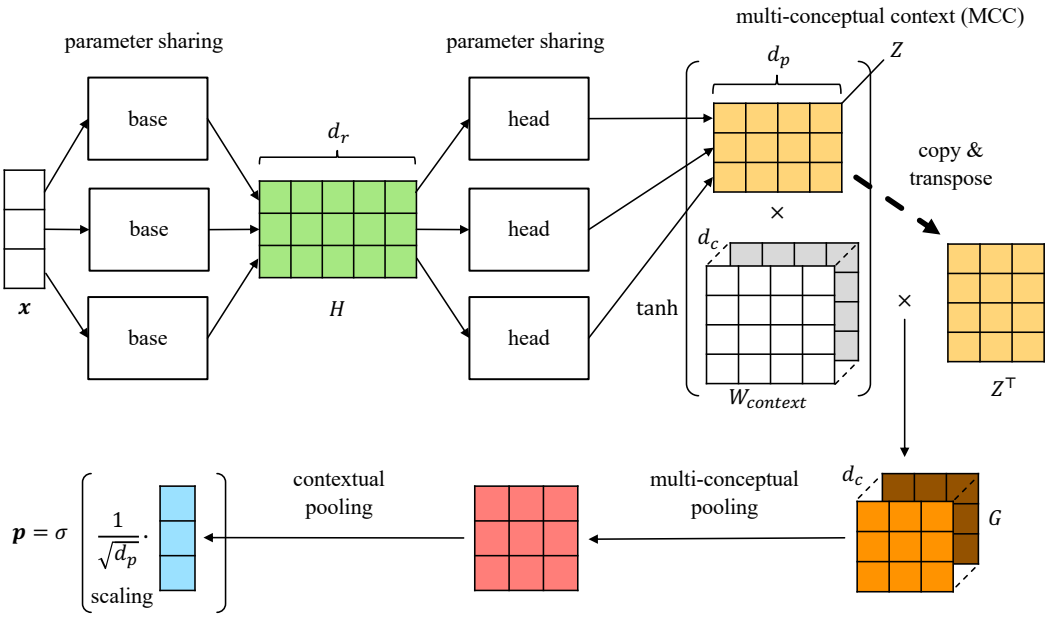


Figure 4-3: Architecture of S^{3}ADNet. The input sequence $\boldsymbol{x} \in \mathbb{R}^{d \times L}$ encoded into a sequential representation $H \in \mathbb{R}^{L \times d_r}$, embedded into a sequential projection $Z \in \mathbb{R}^{L \times d_p}$. Next, the MCC layer acts as the self-attention mechanism to compute the correlations $G \in \mathbb{R}^{L \times d_c \times L}$ among the data points in the sequence. After that, the correlations G are aggregated into a point-to-point negative energy vector. Finally, the sigmoid function activate the scaled vector into the probabilities $\boldsymbol{p} \in (0, 1)^L$.

In S^{3}ADNet, the weight of multi-conceptual context (MCC) $W_{context} \in \mathbb{R}^{d_p \times d_c \times d_p}$ is designed to relate the data points with each other, where d_c is the number of computational concepts. The products in $G = \tanh\left(ZW_{context}\right) Z^\top$ are calculated with the tensor dot operator. It is expected that the MCC weight learns different relationships among the concepts by initializing each matrix of context with random values in a uniform distribution. Besides, since applying the sigmoid function to large values produce results tending to 0 or 1 where the gradients are too small to learn, the negative energy vector is scaled by $1/\sqrt{d_p}$. The anomaly detector consists of MCC,

multi-conceptual pooling, contextual pooling, and the scaled sigmoid activation.

4.3 Algorithms

The algorithm to train the proposed model with the loss functions mentioned above is described as Algorithm 4, 5 and 6.

Algorithm 4: PCL's main learning algorithm for each epoch

Input: training data X, sequence size L, max number of epochs to warm up M, current epoch m, batch size N, temperature T, lookahead size s, penalty α, attemperation τ, data augmentation family Λ, base encoder f, head for agreement g, head for anomaly detection g', anomaly detector ϕ

foreach *minibatch* $\left[\left[x_i^{(k)}\right]_{i=1}^{L}\right]_{k=1}^{N}$ *sampled from* X **do**
  for $k \leftarrow 1$ **to** N **do**
    for $i \leftarrow 1$ **to** L **do**
      draw two augmentation operators $\lambda \sim \Lambda$, $\lambda' \sim \Lambda$
      $\left(\boldsymbol{z}_i^{(2k-1)}, \boldsymbol{z'}_i^{(2k-1)}\right) \leftarrow$ `TowHeadProject(`$x_i^{(k)}, \lambda, f, g, g'$`)`
      $\left(\boldsymbol{z}_i^{(2k)}, \boldsymbol{z'}_i^{(2k)}\right) \leftarrow$ `TowHeadProject(`$x_i^{(k)}, \lambda', f, g, g'$`)`
    if $m > M$ **then**
      $\ell_{CARE}(2k-1) \leftarrow$ `CARELoss(`$\left[\boldsymbol{z'}_i^{(2k-1)}\right]_{i=1}^{L}, s, \alpha, \tau, \phi$`)`
      $\ell_{CARE}(2k) \leftarrow$ `CARELoss(`$\left[\boldsymbol{z'}_i^{(2k)}\right]_{i=1}^{L}, s, \alpha, \tau, \phi$`)`
  for $a \leftarrow 1$ **to** $2N$, $b \leftarrow 1$ **to** $2N$ **do**
    for $i \leftarrow 1$ **to** L **do**
      $c_i^{(a,b)} \leftarrow \text{sim}\left(\boldsymbol{z}_i^{(a)}, \boldsymbol{z}_i^{(b)}\right)$ with Equation 4.2
  define $\ell_{SNTX}(a,b) = -\frac{1}{L}\sum_{i=1}^{L} \log \frac{\exp\left(c_i^{(a,b)}/T\right)}{\sum_{k=1}^{2N} \exp\left(c_i^{(a,b)}/T\right)^{[k \neq a]}}$
  $\mathcal{L}_{SNTX} \leftarrow \sum_{k=1}^{N} \left[\ell_{SNTX}(2k-1, 2k) + \ell_{SNTX}(2k, 2k-1)\right]$
  $\mathcal{L} \leftarrow \mathcal{L}_{SNTX}$
  if $m > M$ **then**
    $\mathcal{L}_{CARE} \leftarrow \sum_{k=1}^{N} \left[\ell_{CARE}(2k-1) + \ell_{CARE}(2k)\right]$
    $\mathcal{L} \leftarrow \mathcal{L} + \mathcal{L}_{CARE}$
  $\mathcal{L} \leftarrow \mathcal{L}/2N$
  update networks f, g, g', and ϕ to minimize $\mathcal{L}$

Algorithm 5: 2-head projection

Function `TowHeadProject`(x, λ, f, g, g')

  Input: data point x, data augmentation λ, base encoder f, head for agreement g, head for anomaly detection g'

  $\tilde{x} \leftarrow \lambda(x)$

  $\boldsymbol{h} \leftarrow f(\tilde{x})$

  $\boldsymbol{z} \leftarrow g(\boldsymbol{h})$

  $\boldsymbol{z'} \leftarrow g'(\boldsymbol{h})$

  return $(\boldsymbol{z}, \boldsymbol{z'})$

Algorithm 6: Context-attempered relative entropy loss

Function `CARELoss`$([\boldsymbol{z}_i]_{i=1}^{L}, s, \alpha, \tau, \phi)$

  Input: projections $[\boldsymbol{z}_i]_{i=1}^{L}$, lookahead size s, penalty α, attemperation τ, anomaly detector ϕ

  $[p_i]_{i=1}^{L} \leftarrow \phi\left([\boldsymbol{z}_i]_{i=1}^{L}\right)$

  for $i \leftarrow 1$ **to** $L-1$ **do**

    for $j \leftarrow i+1$ **to** $\min(i+s, L)$ **do**

      $p_{ij} \leftarrow p_i + p_j - p_i p_j$

      $\bar{p}_{ij} \leftarrow 1 - p_{ij}$

      $r_{ij} \leftarrow \mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau(i, j)$ with Equation 4.2

      $q_{ij} \leftarrow \sigma(r_{ij})$ with Equation 4.6

      $\bar{q}_{ij} \leftarrow 1 - q_{ij}$

      $d_{ij} \leftarrow D_\alpha(p_{ij} \| \bar{q}_{ij})$ with Equation 4.11

  return $\frac{1}{L-1} \sum_{i=1}^{L-1} \frac{1}{\min(s, L-i)} \sum_{j=i+1}^{\min(i+s, L)} d_{ij}$

In the main algorithm, since the parameters of networks are initialized with random values, warming up the training only with $\mathcal{L}_{SNTX}$ makes the encoder learn roughly meaningful representation at the beginning M epochs. After adding $\mathcal{L}_{CARE}$, the anomaly detector starts to be tuned with the coarse representations, and then the probabilities of anomaly occurrence guide the learning of sequential relationships by Equation 4.9.

4.4 Experiments of Outlier and Changepoint Detection on Synthetic Data

Several experiments were performed to verify the PCL assumptions, including outlier detection and changepoint detection on simple one-dimensional synthetic data. The results were compared with SmartSifter [93] and ChangeFinder [94].

4.4.1 Settings for S^{3}ADNet

In the data preprocessing, min-max scaling and z-score normalization were applied to each synthetic dataset. The samples for training were generated by using a sliding window with the size $L = 32$ and the stride size of 1 on the preprocessed data, and the order of the windows was reshuffled at each epoch for random minibatches with the size $N = 32$.

In the data augmentation, a Gaussian noise $x' \sim \mathcal{N}(0, 0.1)$ was added to the data points as $\tilde{x} = x + x'$. The network models were implemented in PyTorch and had identical architecture, as shown in Table B.7. All the multi-conceptual pooling and contextual pooling in the models were the average pooling. The number of concepts was set to 4.

In the contrastive learning, I set the temperature $T = 0.1$ for Equation 4.12, the attemperation function $\tau(i, j) = \ln(j - i + 1)$, and the lookahead size $s = 16$ for Equation 4.9. The loss was optimized by stochastic gradient descent (SGD) [95] with the learning rate of 0.1 and Nesterov momentum [96] with the factor of 0.9. Gradient clipping by the range $[-0.25, 0.25]$ was also employed to avoid too large gradients. Moreover, the maximum epochs for warming-up were 5, and the maximum epochs of the entire training were 50.

4.4.2 Outlier Detection

Outlier detection is a common task to find data points far from the others [97–99]. The function $y = 0.02 + 0.4\sin(5x) + \frac{0.05\cos(50x)}{1+\exp(x)} + x' \sim \mathcal{N}(0, 0.05)$ to generate 2,000 data

points in the domain $[-50, 50)$ was to simulate the problem. Then, 40 of them were uniformly sampled to assign outliers drawn from $\mathcal{U}(-3, 3)$. An example of synthetic data is shown in Figure A-6, which was used to train S^{3}ADNet and the initializing of SmartSifter-SDEM. Then, another dataset was generated by using the same approach for the evaluation.

In the prediction of S^{3}ADNet, data points were taken by a sliding window with the size L and the stride size 1 to give to the model sequentially. If a data point has more than one probability of anomaly, the maximum one was chosen as the predictive result. Generally, the samples with predictive probability greater than 0.5 are considered anomalies. Figure 4-4 shows the results by SmartSifter-SDEM and S^{3}ADNet.

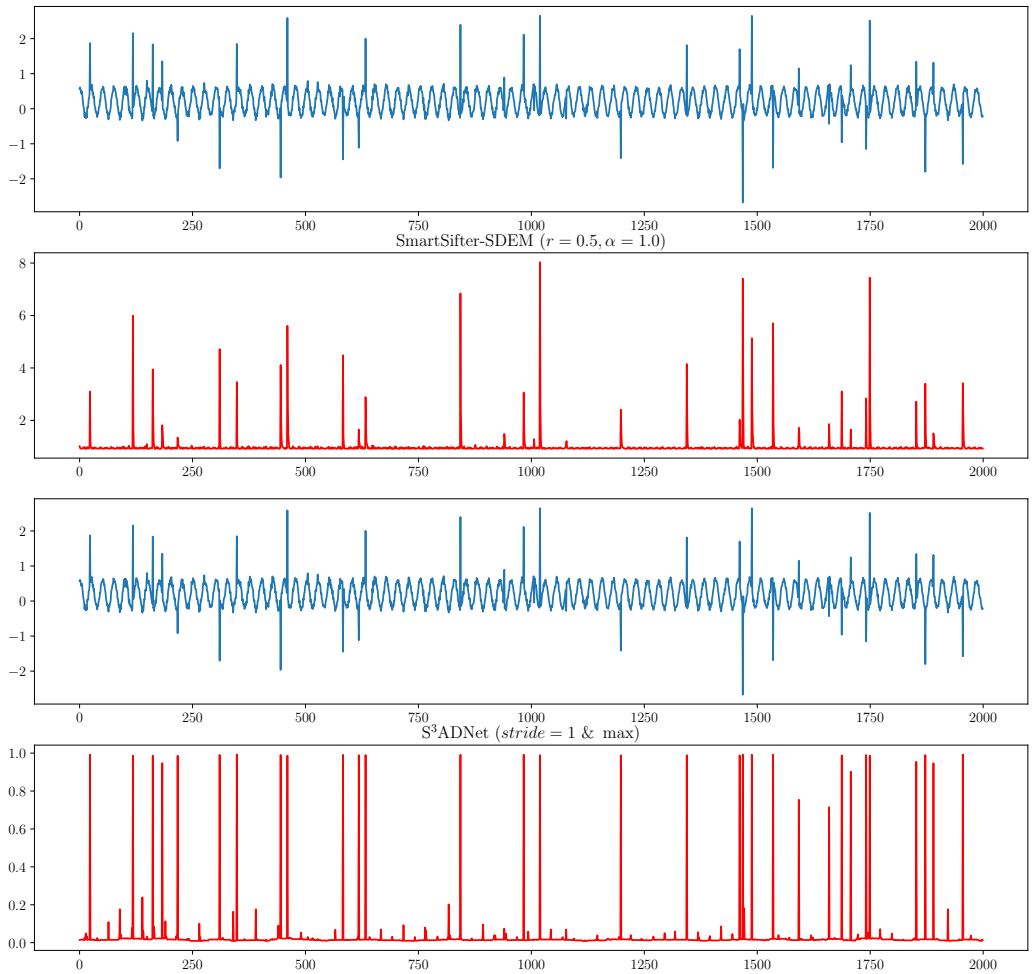


Figure 4-4: Results of outlier detection on synthetic data. The two blue plots are the same evaluation dataset, and the red ones are the outputs from different methods. The parameter r is for the discounting, and α is for the stability in SmartSifter-SDEM.

We found that most of the outliers were detected by S^{3}ADNet with high probabilities from the results. Since the anomaly scores given by SmartSifter were positive real numbers and had such a large variance, it was hard to determine a proper threshold to detect the outliers.

4.4.3 Changepoint Detection

Changepoint detection is to discover abrupt variations in sequential data [100, 101]. Four one-dimensional data segments for each dataset in the experiment were generated to imitate simple situations of change points arising. In the training set for S^{3}ADNet, the data points in the four segments were drawn from $\mathcal{N}(0.7, 0.05)$, $\mathcal{N}(1.5, 0.05)$, $\mathcal{N}(0.6, 0.05)$, and $\mathcal{N}(1.3, 0.05)$ sequentially, and there were 300 points in each segment as shown in Figure A-7 . In the evaluation set, the instances in the segments were sampled from $\mathcal{N}(0, 0.05)$, $\mathcal{N}(2, 0.05)$, $\mathcal{N}(0.2, 0.05)$, and $\mathcal{N}(1.0, 0.05)$ sequentially, and the numbers of data points in them were 350, 100, 300, 150 respectively. Figure 4-5 exhibits the evaluation results by ChangeFinder and S^{3}ADNet.

We observed that ChangeFinder detected incorrect changepoints at the beginning points, and it scarcely discovered the second change point near 450. However, S^{3}ADNet found all the three change points in different stride settings. Furthermore, the sensitivity of S^{3}ADNet's detection changes as adjusting the sliding stride. The smaller stride we set, the higher sensitivity the method has.

Moreover, a combined detection task for evaluation was and operated, as shown in Figure 4-6. It was difficult to distinguish the outliers and change points very well from the results by SmartSifter and ChangeFinder. However, there could be some perturbations around the change points in the outputs by S^{3}ADNet.

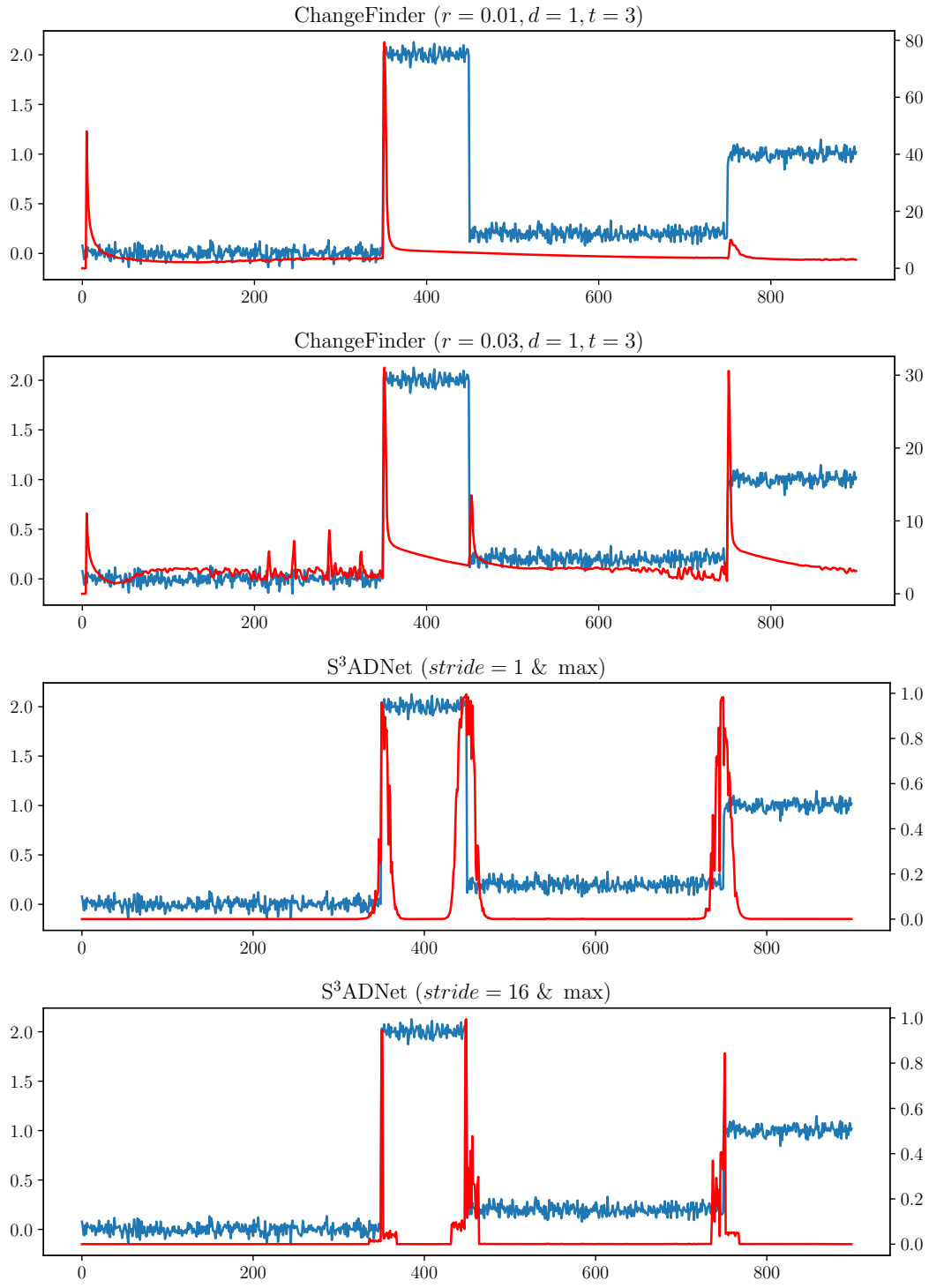


Figure 4-5: Results of changepoint detection on synthetic data. The blue plots are the same evaluation dataset, and the red ones are the outputs from different methods. The parameter r is for the discounting, d is the degree of the autoregression model, and t is the length for moving average in ChangeFinder.

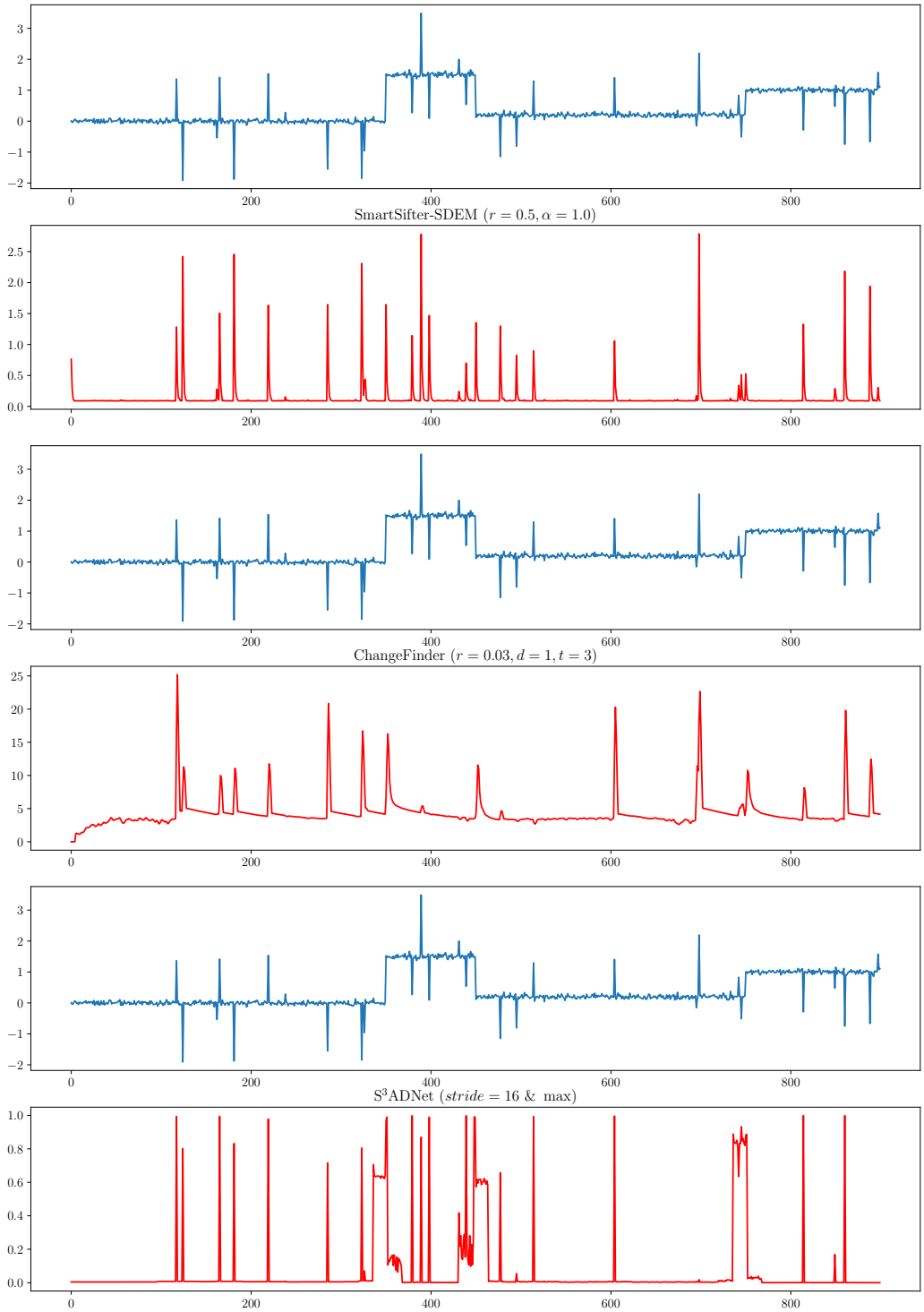


Figure 4-6: Results of changepoint-and-outlier detection on synthetic data. The blue plots are the same evaluation dataset, and the red ones are the outputs from different methods.

4.5 Experiment of Detecting Illegible Handwritten Digits on MNIST

In this experiment, I attempted applying PCL to discover unreadable handwritten digits in the test set of MNIST.

4.5.1 Setups and Result

To build datasets for the task, I grouped the images into ten sequences in the original training and test sets of MNIST by the labels, *i.e.*, the handwritten strokes were labeled to the same digit in each sequence. In the data preprocessing, min-max scaling and z-score normalization were applied as well. The minibatch generation was similar to the one in Section 4.4.1, yet the window size and minibatch size were set as $L = 16, N = 256$. Besides, the orders of images in each sequence were reshuffle at each epoch on the training set.

For the sake of data augmentation, the handwritten images were rotated, translated, scaled, and shared in random ranges, as shown in B.8. Table B.9 represents the architectures of network models for the experiments. Both the multi-conceptual pooling and contextual pooling in the model were the average pooling. The number of concepts was set to 8.

In the training, the temperature was set as $T = 0.1$, the attemperation a constant $\tau(i, j) = 1$, and the lookahead size $s = 16$. SGD was also used to optimize the loss with the learning rate of 0.1 and the momentum of 0.9. Additionally, the number of warm-up epochs was five, and the maximum number of epochs was 250.

With the criterion of the maximum predictive probability greater than 0.5 in sliding windows over the image sequences of the original orders, the result of anomalies found by the trained S^3ADNet model from the test set is shown in Figure 4-7. We can find that most of them are not easy to recognize for us, and the model successfully extracted features from the images to make the prediction. To understand the model's prediction, we can visualize the MCC layer with heatmaps as described in the following

section.

4.5.2 Example of Visualization of Multi-conceptual Context

Take the 3604th image (having the highest probability 0.87 for the label "7") in Figure 4-7 for the example. Figure 4-8a shows the input sequence of the prediction. We can see that the patterns of the 16 handwritten *7* images in the sequence are various, yet most of them are easy to identify as "7".

As described in Section 4.2.2, the MCC layer in S^3ADNet is designed to capture the correlations of data points in a sequence as an attention mechanism. Therefore, we can utilize the information of it as in-model interpretation mentioned in Section 2.5.3. Figure 4-8b demonstrates the heatmaps of context in different computational concepts extracted from the example. We can learn how the data points (on X-axis) rate the others' anomalousness (on Y-axis) from the visualization. For instance, every matrix in Figure 4-8b shows that the twelfth handwritten *7* has a high anomalousness rated by the others.

4.6 Discussion

The above experiments showed that the three assumptions could instruct the deep model in learning representative features and detecting anomalies. However, it is different from the online learning algorithms (*e.g.*, SmartSifter and ChangeFinder) that S^3ADNet needs a certain number of samples to optimize the parameters offline. As well as the prior method SimCLR, PCL has problems in the convergence of loss, significantly affected by our choice of hyperparameters and optimizers. The introduced hyperparameters (*i.e.*, window size, lookahead size, sampling penalty, and attemperation function) may also increase the optimization's difficulty in practices. For example, we attempted employing PCL on the laser machining data as shown in Figure A-9, but the results were highly variant by different hyperparameter settings. Thus, it could be a good idea to collaborate with some AutoML [102] technologies to facilitate the hyperparameter optimization in PCL. Furthermore, all the weight

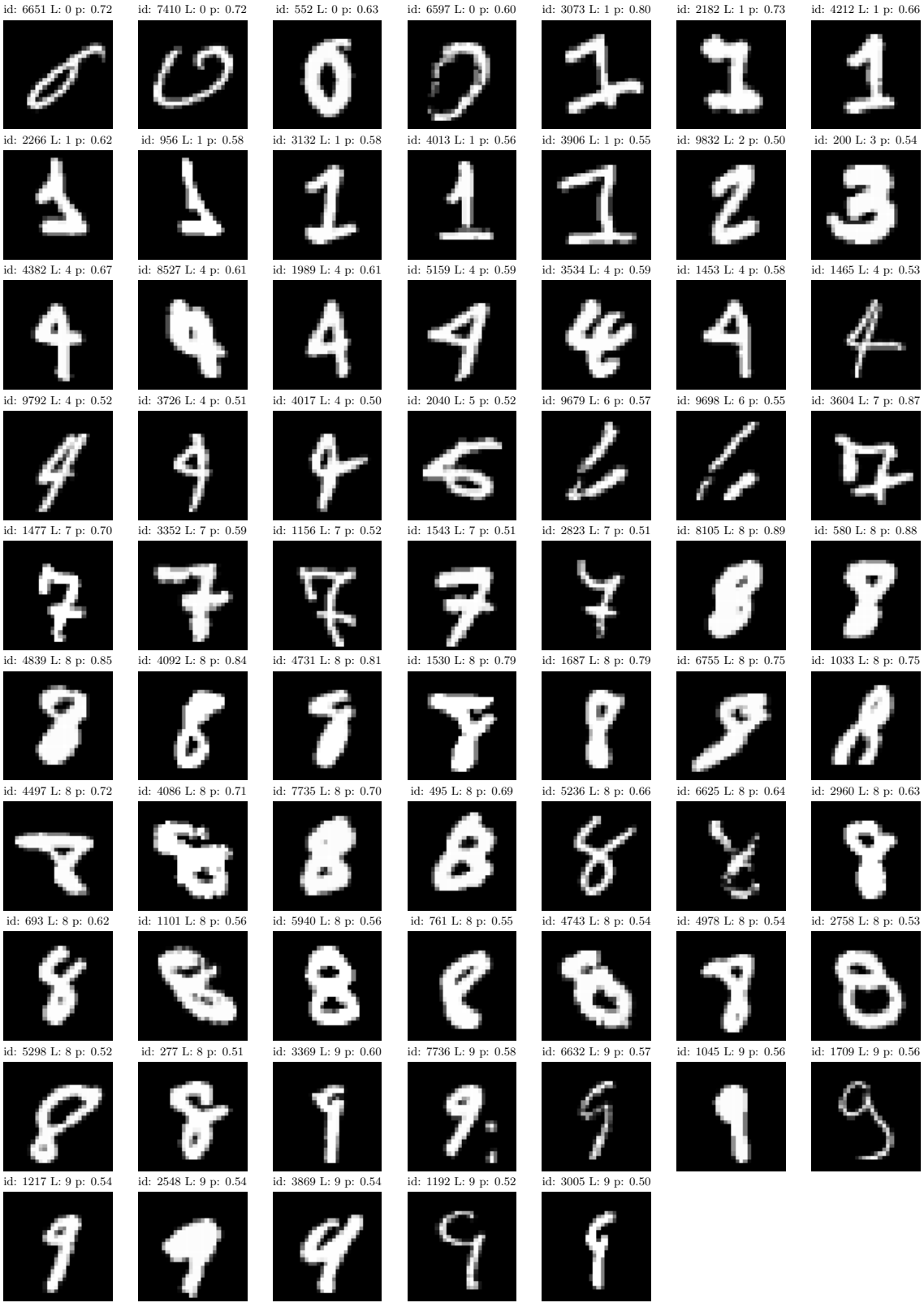
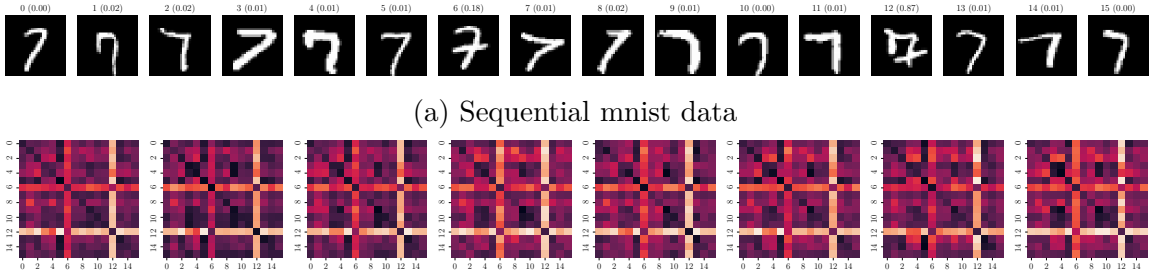


Figure 4-7: Illegible handwritten digits found by S^3*ADNet*. On top of each image, "id" is the image's index in the test set, "L" is the label, and "p" is the predictive probability of anomaly.



(a) Sequential mnist data

(b) Visualization of MCC

Figure 4-8: Example of sequential MNIST data with the prediction of anomaly the visualization of MCC on it. (a) On top of each image, the first number is the index in the sequence, and the bracketed one is the predictive probability. (b) Axes denote the data points with corresponding indices in the sequence. Lighter is higher anomalousness in the heatmaps.

matrices in the MCC layer were trained simultaneously at each iteration in the experiments. We could introduce Dropout to randomly pick up a subset of matrices to train against overfitting [15].

In the experiment of detecting illegible handwritten digits, the prediction was only performed on the test set of the original order. It could be improved by random orders with iterations for the variety of the sample contrasting.

Besides, whether those learned representations have good generalization has not been verified in this study. Since the anomalies are hard to recognize from the raw data for humans, we have to confirm the model's validity via practical applications. Therefore, it requires more evaluations on PCL used for downstream tasks in the future.

4.7 Related Work

Many notable works have been proposed to leverage deep learning to improve anomaly detection performance [103, 104].

Most of the methods for unsupervised or semi-supervised sequential anomaly detection employ RNN with generative adversarial networks (GANs), such as Ano-GAN [105, 106], MAD-GAN [107], and LVEAD [108]; or RNN with variational autoencoders (VAEs), such as RDA [109] and DAE-RNN [110]. Generally, GAN-based

methods require at least one labeled anomaly for the sampling, while VAE-based methods use normal data to train in-distribution models to expose out-distribution data. However, PCL is a non-GAN and non-VAE based method to solve the problem that anomalies can not be easily separated from the data with human work. Also, rather than RNN, the variable attemperation and the MCC layer in PCL are used to learn the sequences' temporal relations.

Chapter 5

Conclusions and Future Works

Data is of the essence for the innovations of the industry and our society. For the sake of effective data utilization, many deep learning methods have been proposed. To solve the interpretability problem of deep models, we introduced new nonlinearized rules to layer-wise relevance propagation as NRP and proposed an energy-based method, KISS, with which encouraging results were obtained in the experiments. The survey we conducted showed that model interpretation could help us reflect upon the deep models and discover novelty. For exploring the application of DL on laser machining data, we executed experiments on the evaluation of deep models and found that the AlexNet-based model in multi-task learning was the better usage on the dataset. The laser machining data is also one kind of sequential data where anomalies are difficult to be labeled due to a lack of prior knowledge. Motivated by that, I designed pessimistic contrastive learning by making the data points in sequences compare with each other to recognize the anomalies with the CARE loss and the S^3ADNet model.

Nevertheless, there are problems and limitations in the proposed methods. Besides the parameter selection problem in NRP and KISS, our MI approach are for local interpretation. If we want to explain the deep models globally, it could be promising that using clustering on the explanatory results grouped by the predictive classes. Since convolutional networks are high-performance for computer vision tasks, we applied CNN to the laser machining data in the evaluation. However, the Fourier speckle patterns are not ordinary objects for the human visual system but complexly

interrelated values. Maybe we can employ some transform-based neural networks (*e.g.*, Fourier neural operator [111] or multi-level wavelet CNN [112]) to the tasks. To improve PCL's training, we could use the mentioned AutoML approach and combine available label information into the multi-task framework for a more constrained objective. Also, we can apply other MI methods to PCL to enhance its interpretability.

Finally, since this thesis focused on developing technologies for the first and second steps of the DL-based knowledge discovery framework, we need to do more work on making guidance and systems to help people better use the visualization results for knowledge discovery in the future.

Appendix A

Figures

Questionnaire (A)

ID: _____

*The questionnaire has (A) and (B) parts. This is the (A) part and has 2 pages.

The (B) part will be sent to you after you finish the (A) part.

In your answers for the following questions, you can use marks (such as  and ) to help your explanation. Also, you can print out the questions and submit your handwritten answers.

(1) What number (0-9) does the picture on the right look like?

Please explain the reason for your answer.

Your answer:

(2) What number (0-9) does the picture on the right look like?

Please explain the reason for your answer.

Your answer:

(3) What number (0-9) does the picture on the right look like?

Please explain the reason for your answer.

Your answer:

Figure A-1: Page 1 of Questionnaire (A).

(4) What does the picture on the right look like?

Please delete the candidates that are not the answer and explain the reason for the answer.

Your answer: *bottlecap* / *tray* / other: _____

(5) What does the picture on the right look like?

Please delete the candidates that are not the answer and explain the reason for the answer.

Your answer: *face powder* / *traffic light* / other: _____

(6) What does the picture on the right look like?

Please delete the candidates that are not the answer and explain the reason for the answer.

Your answer: *beer bottle* / *shield* / other: _____

Figure A-2: Page 2 of Questionnaire (A).

Questionnaire (B)

ID: _____

*The questionnaire has (A) and (B) parts. This is the (B) part and has 2 pages.

To the pictures you have seen in the (A) part, a computer gave its answers as well. It tried explaining the answers with groups of annotated pictures. In each annotation group, the leftmost is the original picture and the rest is the candidate answers with titles. In each title, the bracketed value is the *likeness* considered by the computer for the corresponding candidate (larger is more similar). The annotations colored red/blue mean the supported/unsupported regions for the candidate answers.

In your answers for the following questions, you can use marks (such as  and ) to help your explanation. Also, you can print out the questions and submit your handwritten answers.

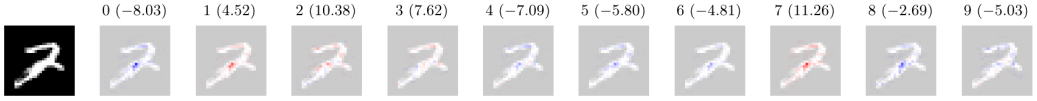
(1) The computer says picture (1) looks more like the number 4. Do you agree with it? Please write your new answer if it is changed and explain the reason for your agreement or disagreement.

Your answer:

(2) The computer says picture (2) looks more like the number 4. Do you agree with it? Please write your new answer if it is changed and explain the reason for your agreement or disagreement.

Your answer:

(3) The computer says picture (3) looks more like the number 7. Do you agree with it? Please write your new answer if it is changed and explain the reason for your agreement or disagreement.



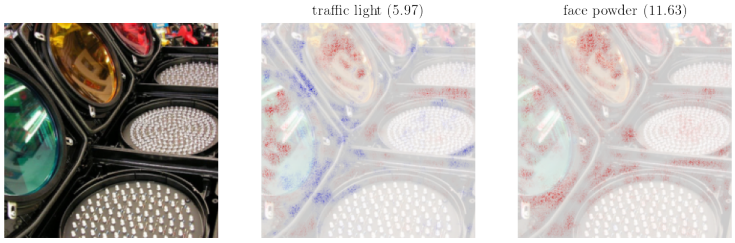
Your answer:

Figure A-3: Page 1 of Questionnaire (B).

(4) The computer says picture (4) looks more like a *bottlecap*. Do you agree with it? Please write your new answer if it is changed and explain the reason for your agreement or disagreement.

Your answer:

(5) The computer says picture (5) looks more like *face powder*s. Do you agree with it? Please write your new answer if it is changed and explain the reason for your agreement or disagreement.



Your answer:

(6) The computer says picture (6) looks more like a *shield*. Do you agree with it? Please write your new answer if it is changed and explain the reason for your agreement or disagreement.

Your answer:

Figure A-4: Page 2 of Questionnaire (B).

Figure A-5: Example of KISS on the AlexNet-based model on laser machining data ($|S| = 5, l = 5$ with uniform sampling).

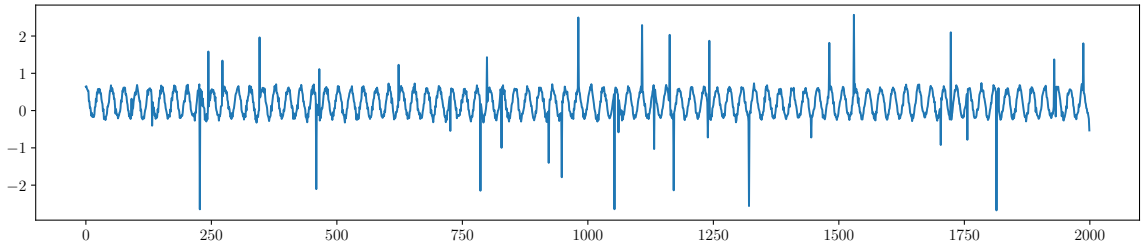


Figure A-6: Example of synthetic data with outliers.

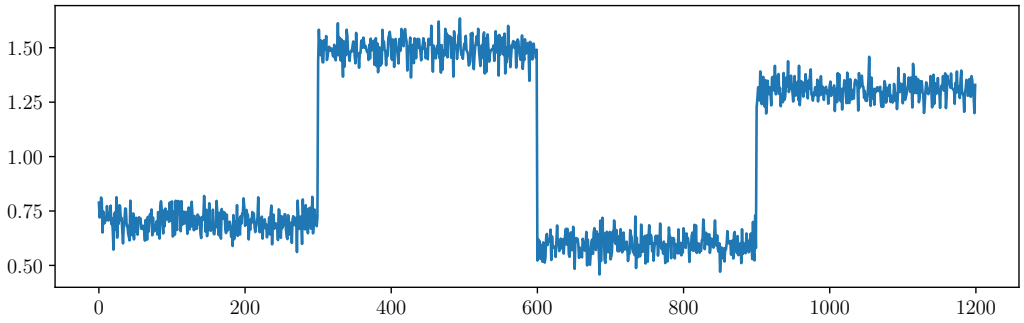


Figure A-7: Example of synthetic data with change points.

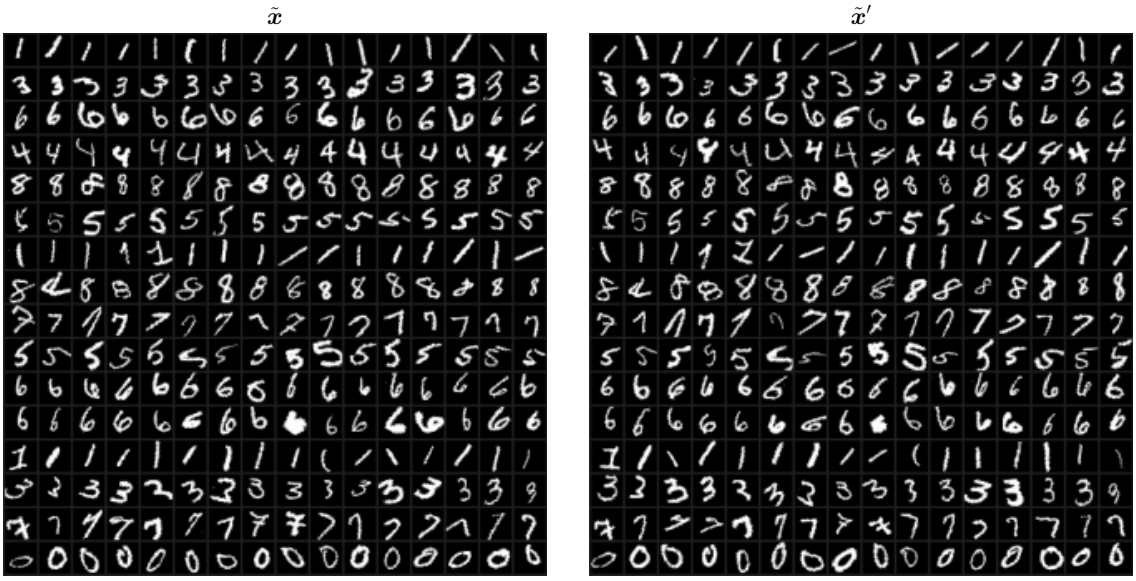


Figure A-8: Example of augmented data on sequential MNIST. The left samples and the right ones were transformed from the same minibatch. Each row is a sequence of some digit label.

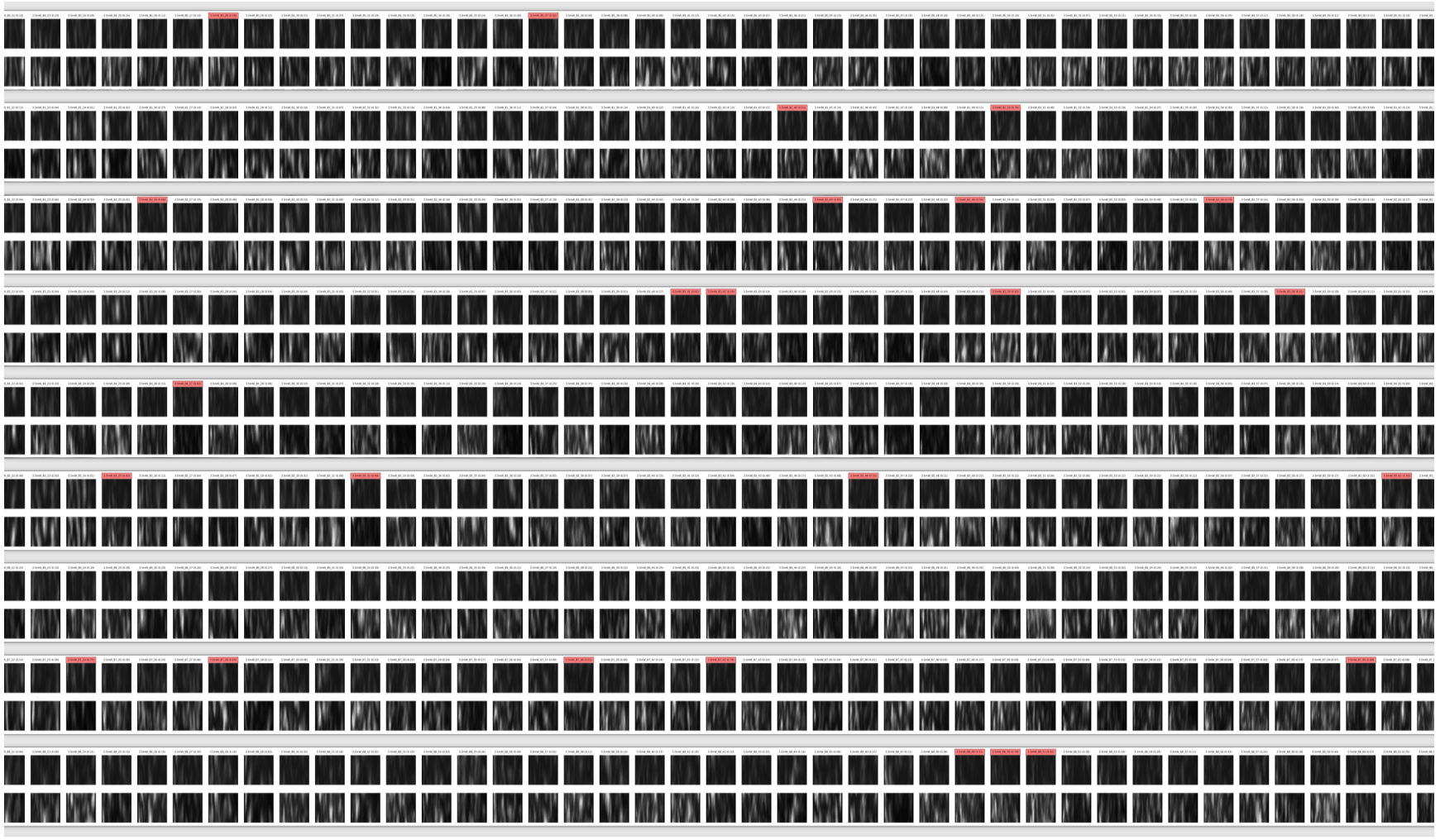


Figure A-9: Example of PCL on laser machining data. Every two rows is a part of a sequence, where the upper shows raw images and the lower shows the self-normalized ones. The images with red titles are predicted as anomalies ($p_i > 0.5$).

Appendix B

Tables

Table B.1: Hyper-parameters and evaluation results of the selected APN models.

Parameters or States	*Computer&Internet*	*Health*	*Society&Culture*
Margin		2	
Dropout probability		0.5	
Mini-batch size		16	
Weight decay		1.0×10^{-5}	
Hidden size of LSTM	100	120	100
Epoch	6th	6th	15th
Acc. on dev. (#samples)	0.6015 (1222)	0.4626 (1070)	0.3445 (865)
Acc. on test (#samples)	0.5724 (1223)	0.4654 (1070)	0.2763 (865)

Table B.2: Sequential architecture of the MNIST model in the experiment. The top module is for input data, and the bottom one outputs negative energies.

Module	**Input**	**Output**	**Kernel**	**Stride**	**Activation, *etc.***
Conv2d	1	32	3×3	1×1	ReLU
Conv2d	32	64	3×3	1×1	ReLU
MaxPool			2×2	2×2	Dropout(0.25), Flatten
Linear	9216	128			ReLU, Dropout(0.5)
Linear	128	10			

Table B.3: Answer patterns from 31 respondents. The hesitative answers are represented in the ·/· form.

Question	Label	Top-1 prediction	Answer in (A)	Answer in (B)	Count
(1)	4	4	4	4	22
			9	9	8
			9	4	1
(2)	9	4	9	9	28
			4/9	4	1
			9/4	9	1
			9	4/9	1
(3)	2	7	7	7	17
			2	2	12
			2	7	1
			7	2	1
(4)	tray	bottlecap	bottlecap	bottlecap	15
			tray	tray	11
			ice cream lid	ice cream lid	1
			bottlecap/tray	bottlecap/tray	1
			tray	bottlecap	1
			bottlecap	tray	1
			frisbee	bottlecap	1
(5)	traffic light	face powder	traffic light	traffic light	25
			traffic light	face powder	2
			pearl	traffic light	1
			face powder	face powder	1
			jewel	face powder	1
			music arcade	traffic light	1
(6)	beer bottle	shield	beer bottle	beer bottle	28
			beer can	beer can	2
			shield	shield	1

Table B.4: Explanation patterns in Questionnaire (A) from 31 respondents. A respondent could reply with more than one reason in the explanation.

Question	Reason	Count
(1)	Sharp corners	14
	N/A	5
	No right protrusion	3
	Positional relationship	3
	By habit	2
	By intuition	2
	Enclosure's size	1
	Like a fix from 4	1
(2)	Left round corner	8
	N/A	6
	By habit	4
	By intuition	3
	Stroke order	3
	One stroke	3
	Positional relationship	2
	Seen writing way	2
	Upper right Hole	1
(3)	By intuition	8
	Middle dash	5
	N/A	4
	Seen writing way	3
	By habit	3
	Stroke order	2
	First stroke	2
	Lower left loop	2
	Last stroke	1

Question	Reason	Count
(4)	Smooth rim	7
	By intuition	5
	The text	5
	Concave shape	5
	N/A	4
	Circle shape	2
	Convex shape	2
	The logo	1
(5)	Three colors	17
	By intuition	5
	LEDs	5
	N/A	3
	Inner structure	2
	The size	1
(6)	The text	14
	The brand	8
	N/A	4
	The logo	3
	By intuition	1
	Surface's curve	1
	Cylindrical shape	1

Table B.5: Explanation patterns in Questionnaire (B) from 31 respondents, where NAND denotes "neither agree nor disagree". A respondent could reply with more than one reason in the explanation.

Question	Agreement	Reason	Count
(1)	Y	N/A	14
		Left sharp corner	9
	N	No right protrusion	4
		Bended vertical line	2
		N/A	1
		Positional relationship	1
(2)	N	Left round corner	6
		No right protrusion	6
		By intuition	4
		Upper right hole	3
		One stroke	3
		By habit	2
		N/A	2
		Last stroke	1
		Never seen	1
		Positional relationship	1
	Y	N/A	1
	NAND	Not sure	1
(3)	Y	N/A	11
		Middle dash	5
		By habit	1
	N	Lower left loop	5
		Last stroke	3
		Writing way	2

		By intuition	2
		The tilt	1
		Horizontal lines	1
(4)	Y	N/A	11
		Convex shape	2
		The text	2
		Concave shape	2
		Not sure	1
	N	Smooth rim	8
		Concave shape	2
		N/A	1
		The structure	1
	NAND	N/A	1
(5)	N	Three colors	10
		Inner structure	7
		The material	7
		N/A	4
		LEDs	3
	Y	N/A	2
		Black rims	1
		Seen before	1
(6)	N	The text	15
		The brand	4
		Cylindrical shape	4
		N/A	3
		By intuition	2
		The material	1

	Y	Surface's curve	1

Table B.6: Splits of the laser machining dataset.

Subset name	**Range of experiment IDs**	**Number of samples**
Training	1–70	175,000
Validation	71–85	37,500
Test	86–105	50,000

Table B.7: Architecture of the S^3ADNet model on synthetic data. The Identity modules output the same values as the input without parameters, *i.e.*, the two heads used the output of the base encoder as the projections.

Module	**Module**	**Input**	**Output**	**Activation**
Base encoder	Linear	1	16	Tanh
	Linear	16	16	Tanh
	Linear	16	8	
Embedding head	Identity	8	8	
Detection head	Identity	8	8	

Table B.8: Data augmentation on sequential MNIST. PyTorch's RandomAffine module was used as the image transformer.

Operator	**Random range**
rotate	$[-30°, 30°]$
translate	$[-3, 3]$
scale	$[0.8, 1.2]$
shear	$[-20°, 20°]$

Table B.9: Architecture of the S^3ADNet model on sequential MNIST.

Network	**Module**	**Input**	**Output**	**Kernel**	**Stride**	**Activation, *etc.***
Base	Conv2d	1	32	5×5	2×2	LeakyReLU(0.1)
	Conv2d	32	64	3×3	1×1	LeakyReLU(0.1)
	MaxPool2d			2×2	2×2	Dropout(0.25), Flatten
	Linear	1600	128			LeakyReLU(0.1)
	BatchNorm	128	128			Dropout(0.5)
Embedding	Linear	128	128			
Detection	Linear	128	128			

Appendix C

Publications

C.1 Journal Articles (Peer Reviewed)

- Quexuan Zhang, Zexuan Wang, Bin Wang, Yukio Ohsawa, and Teruaki Hayashi. Feature extraction of laser machining data by using deep multi-task learning. *Information* 11, no. 8: 378, 2020.

C.2 Conference Articles (Peer Reviewed)

- ◯Quexuan Zhang and Yukio Ohsawa. Kiss: an ebm-based approach for explaining deep models. In *Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24th International Conference KES2020, Procedia Computer Science*, volume 176, pages 271–280. Elsevier, 2020. Verona, Italy (virtual).

- ◯Quexuan Zhang and Yukio Ohsawa. Nonlinearized relevance propagation. In *Pacific Rim International Conference on Artificial Intelligence*, pages 904–914. Springer, 2018. Nanjing, China.

Bibliography

[1] Klaus Schwab. *The fourth industrial revolution*. Currency, 2017.

[2] Mayumi Fukuyama. Society 5.0: Aiming for a new human-centered society. *Japan Spotlight*, 27:47–50, 2018.

[3] Oded Maimon and Lior Rokach. *Data mining and knowledge discovery handbook*. Springer, 2005.

[4] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*. MIT press Cambridge, 2016.

[5] Andreas Kamilaris and Francesc X Prenafeta-Boldú. Deep learning in agriculture: A survey. *Computers and electronics in agriculture*, 147:70–90, 2018.

[6] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020.

[7] Kalaivani Sundararajan and Damon L Woodard. Deep learning for biometrics: A survey. *ACM Computing Surveys (CSUR)*, 51(3):1–34, 2018.

[8] Wen Long, Zhichen Lu, and Lingxiao Cui. Deep learning-based feature engineering for stock price movement prediction. *Knowledge-Based Systems*, 164:163–173, 2019.

[9] Dan Guest, Kyle Cranmer, and Daniel Whiteson. Deep learning and its application to lhc physics. *Annual Review of Nuclear and Particle Science*, 68:161–181, 2018.

[10] Kurt Hornik, Maxwell Stinchcombe, Halbert White, et al. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[11] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[12] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence

(xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

[13] Md Zahangir Alom, Tarek M Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C Van Esesn, Abdul A S Awwal, and Vijayan K Asari. The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

[14] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[15] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[16] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

[17] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[18] Yukio Ohsawa, Hiroyuki Kido, Teruaki Hayashi, Chang Liu, and Kazuhiro Komoda. Innovators marketplace on data jackets, for valuating, sharing, and synthesizing data. In *Knowledge-Based Information Systems in Practice*, pages 83–97. Springer, 2015.

[19] Yukio Ohsawa, Nels E Benson, and Masahiko Yachida. Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. In *Proceedings IEEE International Forum on Research and Technology Advances in Digital Libraries-ADL'98-*, pages 12–18. IEEE, 1998.

[20] Shuntaro Tani, Yutsuki Aoyagi, and Yohei Kobayashi. Neural-network-assisted in situ processing monitoring by speckle pattern observation. *Optics Express*, 28(18):26180–26188, 2020.

[21] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[22] Nadeem N Rather, Chintan O Patel, and Sharib A Khan. Using deep learning towards biomedical knowledge discovery. *Int. J. Math. Sci. Comput.(IJMSC)*, 3(2):1–10, 2017.

[23] Tian-en Huang, Qinglai Guo, Hongbin Sun, Chin-Woo Tan, and Tianyu Hu. A deep learning approach for power system knowledge discovery based on multitask learning. *IET Generation, Transmission & Distribution*, 13(5):733–740, 2018.

[24] Bo Xu, Yong Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Wanyun Cui, and Yanghua Xiao. Cn-dbpedia: A never-ending chinese knowledge extraction system. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 428–438. Springer, 2017.

[25] Fadil Santosa and William W Symes. Linear inversion of band-limited reflection seismograms. *SIAM Journal on Scientific and Statistical Computing*, 7(4):1307–1330, 1986.

[26] John Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[27] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, 19(6):1236–1246, 2018.

[28] Jinkyu Kim and John Canny. Interpretable learning for self-driving cars by visualizing causal attention. In *Proceedings of the IEEE international conference on computer vision*, pages 2942–2950, 2017.

[29] Christoph Molnar. *Interpretable Machine Learning*. Lulu. com, 2020.

[30] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[31] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

[32] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

[33] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[34] Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. *arXiv preprint arXiv:1702.04595*, 2017.

[35] Brandon Carter, Jonas Mueller, Siddhartha Jain, and David Gifford. What made you do this? understanding black-box decisions with sufficient input subsets. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 567–576. PMLR, 2019.

[36] Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. Learning to explain: An information-theoretic perspective on model interpretation. *arXiv preprint arXiv:1802.07814*, 2018.

[37] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[38] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, pages 9505–9515, 2018.

[39] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[40] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. *arXiv preprint arXiv:1704.02685*, 2017.

[41] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[42] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[43] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[44] Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. Explaining recurrent neural network predictions in sentiment analysis. *arXiv preprint arXiv:1706.07206*, 2017.

[45] Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, 2017.

[46] Quexuan Zhang and Yukio Ohsawa. Nonlinearized relevance propagation. In *Pacific Rim International Conference on Artificial Intelligence*, pages 904–914. Springer, 2018.

[47] Ming Tan, Cicero Dos Santos, Bing Xiang, and Bowen Zhou. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 464–473, 2016.

[48] Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. Grammar as a foreign language. In *Advances in neural information processing systems*, pages 2773–2781, 2015.

[49] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[50] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[51] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[52] Quexuan Zhang and Yukio Ohsawa. Kiss: an ebm-based approach for explaining deep models. *Procedia Computer Science*, 176:271–280, 2020.

[53] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[54] Charles Kittel. *Elementary statistical physics*. Courier Corporation, 2004.

[55] Pavlos S Efraimidis and Paul G Spirakis. Weighted random sampling with a reservoir. *Information Processing Letters*, 97(5):181–185, 2006.

[56] Kim-Hung Li. Reservoir-sampling algorithms of time complexity o (n (1+ log (n/n))). *ACM Transactions on Mathematical Software (TOMS)*, 20(4):481–493, 1994.

[57] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[58] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[59] Pieter-Jan Kindermans, Kristof Schütt, Klaus-Robert Müller, and Sven Dähne. Investigating the influence of noise and distractors on the interpretation of neural networks. *arXiv preprint arXiv:1611.07270*, 2016.

[60] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*, 2017.

[61] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[62] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[63] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*, 2017.

[64] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[65] Erico Tjoa and Cuntai Guan. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[66] Amitojdeep Singh, Sourya Sengupta, and Vasudevan Lakshminarayanan. Explainable deep learning models in medical image analysis. *arXiv preprint arXiv:2005.13799*, 2020.

[67] Chengliang Yang, Anand Rangarajan, and Sanjay Ranka. Global model interpretation via recursive partitioning. In *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 1563–1570. IEEE, 2018.

[68] Andrew Zupon, Maria Alexeeva, Marco Valenzuela-Escárcega, Ajay Nagesh, and Mihai Surdeanu. Lightly-supervised representation learning with global interpretability. In *Proceedings of the Third Workshop on Structured Prediction for NLP*, pages 18–28, 2019.

[69] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[70] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[71] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[72] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pages 3504–3512, 2016.

[73] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*, 2019.

[74] George Chryssolouris, Panagiotis Stavropoulos, and Konstantinos Salonitis. *Process of Laser Machining*, chapter VII, pages 1601–1628. Springer, London, 2013.

[75] Syed AM Tofail, Elias P Koumoulos, Amit Bandyopadhyay, Susmita Bose, Lisa O'Donoghue, and Costas Charitidis. Additive manufacturing: scientific and technological challenges, market uptake and opportunities. *Materials today*, 21(1):22–37, 2018.

[76] Lorenzo Bassi. Industry 4.0: Hope, hype or revolution? In *2017 IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI)*, pages 1–6. IEEE, 2017.

[77] Bruno Salgues. *Society 5.0: Industry of the Future, Technologies, Methods and Tools*. John Wiley & Sons, 2018.

[78] Yoshihisa Yamamoto, Masahide Sasaki, and Hiroki Takesue. Quantum information science and technology in japan. *Quantum Science and Technology*, 4(2):020502, 2019.

[79] Quexuan Zhang, Zexuan Wang, Bin Wang, Yukio Ohsawa, and Teruaki Hayashi. Feature extraction of laser machining data by using deep multi-task learning. *Information*, 11(8):378, 2020.

[80] Jan J Gerbrands. On the relationships between svd, klt and pca. *Pattern recognition*, 14(1-6):375–381, 1981.

[81] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[82] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[83] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[84] Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472. IEEE, 2017.

[85] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[86] Per Christian Hansen. The truncatedsvd as a method for regularization. *BIT Numerical Mathematics*, 27(4):534–553, 1987.

[87] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[88] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *Advances in Neural Information Processing Systems*, pages 15663–15674, 2019.

[89] Ben Mills, Daniel J Heath, James A Grant-Jacob, Yunhui Xie, and Robert W Eason. Image-based monitoring of femtosecond laser machining via a neural network. *Journal of Physics: Photonics*, 1(1):015008, 2018.

[90] Jack Francis and Linkan Bian. Deep learning for distortion prediction in laser-based additive manufacturing using big data. *Manufacturing Letters*, 20:10–14, 2019.

[91] MDT McDonnell, JA Grant-Jacob, Y Xie, M Praeger, BS Mackay, RW Eason, and B Mills. Modelling laser machining of nickel with spatially shaped three pulse sequences using deep learning. *Optics Express*, 28(10):14627–14637, 2020.

[92] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[93] Kenji Yamanishi, Jun-Ichi Takeuchi, Graham Williams, and Peter Milne. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Mining and Knowledge Discovery*, 8(3):275–300, 2004.

[94] Kenji Yamanishi, Junnichi Takeuchi, and Yuko Maruyama. Data mining for security. *NEC journal of advanced technology*, 2(1):63–69, 2005.

[95] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[96] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[97] Charu C Aggarwal and Philip S Yu. Outlier detection for high dimensional data. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 37–46, 2001.

[98] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126, 2004.

[99] Peter J Rousseeuw and Annick M Leroy. *Robust regression and outlier detection*, volume 589. John wiley & sons, 2005.

[100] Robert Lund, Xiaolan L Wang, Qi Qi Lu, Jaxk Reeves, Colin Gallagher, and Yang Feng. Changepoint detection in periodic and autocorrelated time series. *Journal of Climate*, 20(20):5178–5190, 2007.

[101] Samaneh Aminikhanghahi and Diane J Cook. A survey of methods for time series change point detection. *Knowledge and information systems*, 51(2):339–367, 2017.

[102] Xin He, Kaiyong Zhao, and Xiaowen Chu. Automl: A survey of the state-of-the-art. *arXiv preprint arXiv:1908.00709*, 2019.

[103] Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*, 2019.

[104] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. Deep learning for anomaly detection: A review. *arXiv preprint arXiv:2007.02500*, 2020.

[105] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pages 146–157. Springer, 2017.

[106] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, 54:30–44, 2019.

[107] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *International Conference on Artificial Neural Networks*, pages 703–716. Springer, 2019.

[108] Chunkai Zhang and Yingyang Chen. Time series anomaly detection with variational autoencoders. *arXiv preprint arXiv:1907.01702*, 2019.

[109] Chong Zhou and Randy C Paffenroth. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 665–674, 2017.

[110] Weining Lu, Yu Cheng, Cao Xiao, Shiyu Chang, Shuai Huang, Bin Liang, and Thomas Huang. Unsupervised sequential outlier detection with deep architectures. *IEEE transactions on image processing*, 26(9):4321–4330, 2017.

[111] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

[112] Masataka Ichikawa, Tomohisa Sujino, and Takanori Kanai. The relationship between gut microbiome, immune system, and cancer. *Gan to kagaku ryoho. Cancer & chemotherapy*, 46(12):1807–1813, 2019.