information

Article

Decentralized Federated Learning-Enabled Relation Aggregation for Anomaly Detection

Siyue Shuai [1], Zehao Hu [1], Bin Zhang [2], Hannan Bin Liaqat [3] and Xiangjie Kong [1,*]

[1] College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China; 111122120001@zjut.edu.cn (S.S.); 211122120078@zjut.edu.cn (Z.H.)
[2] College of Digital Commerce, Zhejiang Yuexiu University of Foreign Language, Shaoxing 312000, China; 20001008@zyufl.edu.cn
[3] IT Department of Information Sciences, Division of Science & Technology, Township Campus, University of Education, Lahore 54000, Pakistan; hannan.liaqat@ue.edu.pk
* Correspondence: xjkong@zjut.edu.cn; Tel.: +86-158-4090-1926

Abstract: Anomaly detection plays a crucial role in data security and risk management across various domains, such as financial insurance security, medical image recognition, and Internet of Things (IoT) device management. Researchers rely on machine learning to address potential threats in order to enhance data security. In the financial insurance industry, enterprises tend to leverage the relation mining capabilities of knowledge graph embedding (KGE) for anomaly detection. However, auto insurance fraud labeling strongly relies on manual labeling by experts. The efficiency and cost issues of labeling make auto insurance fraud detection still a small-sample detection challenge. Existing schemes, such as migration learning and data augmentation methods, are susceptible to local characteristics, leading to their poor generalization performance. To improve its generalization, the recently emerging Decentralized Federated Learning (DFL) framework provides new ideas for mining more frauds through the joint cooperation of companies. Based on DFL, we propose a federated framework named DFLR for relation embedding aggregation. This framework trains the private KGE of auto insurance companies on the client locally and dynamically selects servers for relation aggregation with the aim of privacy protection. Finally, we validate the effectiveness of our proposed DFLR on a real auto insurance dataset. And the results show that the cooperative approach provided by DFLR improves the client's ability to detect auto insurance fraud compared to single client training.

Keywords: Decentralized Federated Learning; knowledge graph embedding; anomaly detection; relation aggregation

Citation: Shuai, S.; Hu, Z.; Zhang, B.; Liaqat, H.B.; Kong, X. Decentralized Federated Learning-Enabled Relation Aggregation for Anomaly Detection. *Information* **2023**, *14*, 647. https://doi.org/10.3390/info14120647

Academic Editor: Ken McGarry

Received: 7 October 2023
Revised: 25 November 2023
Accepted: 30 November 2023
Published: 3 December 2023

1. Introduction

The development of Internet technology makes digitized data and information easy to be transmitted and analyzed, and the subtle connections between data and information are easier to mine [1]. But at the same time, hidden crises and potential risks, such as abnormal data and fraudulent behavior, are also mixed in. Whether it is fraud detection in the financial field, device quality monitoring in the IoT industry, disease diagnosis in the healthcare field, or intrusion detection in network security, all rely on anomaly detection to ensure system reliability and data integrity. Regardless of the industry sector, all involve serious economic losses and trust crises. Therefore, the research and development of effective detection mechanisms for the management and analysis of digitized information have become crucial.

However, anomaly samples are still rare and traditional auto insurance fraud detection relies directly on expert manual review. This results in extremely inefficient fraud detection [2]. To reduce human error and missed inspections, insurance companies start to

leverage the automated intelligence of machine learning [3]. In addition to using unsupervised learning, semi-supervised learning, and other methods to improve the performance of anomaly detection models, researchers also use these methods: (1) Data generation: generating abnormal data through transformation, expansion, or learning from existing data for sample expansion, such as how Zhang et al. utilized MetaGAN, based on a Generative Adversarial Network, to generate images to strengthen the performance of sample-level image classification [4]. (2) Transfer Learning [5]: learning anomaly detection models from the original domain dataset and transferring them to the target domain. (3) Active Learning [6]: improving model performance by intelligently selecting which samples should be labeled, thereby reducing dependence on labeled data. (4) Cooperative train [7]: Collaborating between different data holders to jointly build anomaly detection models can also help solve the problem of data scarcity. The centralized training of data can indeed improve the efficiency of detection, but this completely disregards privacy concerns [8]. Especially in the financial insurance industry, when it comes to a substantial amount of customer information, data sharing needs to be carried out with the precondition of ensuring privacy protection.

In the insurance industry, with the rapid increase in the number of auto insurance motor vehicles and the swift development of the auto insurance industry, auto insurance fraud has become one of the biggest threats to the current insurance industry [9]. There are numerous ways to commit fraud, including faking the scene of a car accident, using fake license plates, and combining with repair shops to exaggerate damages [10]. According to the annual property and casualty insurance claims service report [11] released by the China Life Insurance (Group) Company in 2022, annual claims amounted to 59.23 billion CNY, of which 40.79 billion CNY was paid out for auto insurance.

For the struggling auto insurance industry, cracking down on fraud is urgent. In order to protect the legitimate rights and interests of consumers and insurance companies, different countries have formulated a large number of rules and regulations. For instance, the "Regulations on Compulsory Insurance for Motor Vehicle Liability Insurance" issued by China provide a basic basis for the handling of insurance disputes in all aspects from insurance coverage and compensation to penalties. To actively fight against auto insurance fraud, the United States has set up the National Insurance Crime Bureau, armed with a series of bills such as the "Pre-Claims Underwriting Inspection Act" and the "Motor Vehicle Claims Information Collection Act".

In addition to solidifying the institutional foundation and strengthening the construction of the insurance industry team, various companies are promoting the in-depth development of anti-insurance-fraud work with technology-enabled fraud detection [12]. The means evolve from traditional insurance detection methods to technology-assisted efficient detection ones, empowered by the following aspects:

- **Reduce over-reliance on high-quality expert experience.** Considering saving labor and time costs in the loss determination, pricing, and compensation process, these companies construct expert knowledge-based empirical information bases in conjunction with image recognition technology to help identify damaged parts, types of damage, and assess the extent of damage.
- **Improve detection efficiency using machine learning.** Insurance companies have extremely high requirements for the timeliness of auto insurance claims. In order to ensure the high accuracy of fraud detection, the insurance company mines the features of historical cases and utilizes machine learning algorithms such as decision tree [13], random forests [14], logistic regression [15], and XGBOOST [16,17] to train AI models that can be applied to reporting and surveying.
- **Utilize graph structure to mine team fraud.** Due to the characteristics of team-based and industrialized insurance fraud in the current auto insurance industry, inspectors may face challenges in swiftly unraveling the implicit connections of team fraud between various cases. Therefore, companies leverage the network structure of graphs to mine additional feature factors from perspectives such as the incident location and

reporting time. By identifying abnormal nodes, edges, and neighboring combinations, they aim to enhance the efficiency and success rate of team fraud detection [18].

We consider adopting collaborative learning to jointly combat fraud, with privacy protection requirements and limitations. However, as shown in Figure 1, the reality is that auto insurance data are usually distributed among different insurance companies and organizations. Since the data contain the privacy of the policyholder's personal information and company confidentiality, it is quite challenging for companies to put into effect traditional centralized data inspection methods in the real world. The auto insurance industry is urgently seeking an inspection method that balances the depth of cooperation with the efficiency of inspection and protects data privacy.

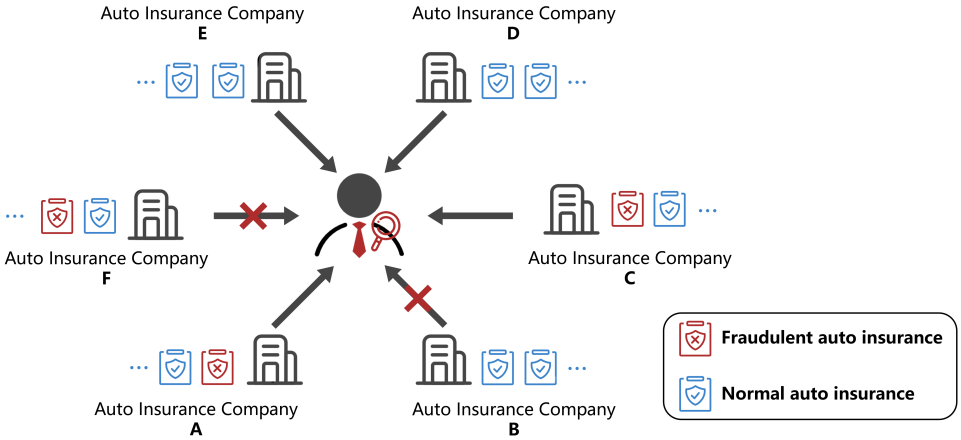

Figure 1. Centralized auto fraud detection dilemma.

The complex relations between auto insurance data naturally draw researchers' attention to the ability of Knowledge Graph (KG) to mine hidden relations [19]. At the same time, the rise of Decentralized Federated Learning provides a new paradigm for data protection [20]. Inspired by this, our work aims to leverage the privacy protection advantages of DFL combined with the high-quality expression of KGE. Our proposed DFL framework, DFLR, is a new solution designed for auto insurance anomalous fraud detection. The contributions of our work are summarized as follows:

- We propose a framework, Decentralized Federated Learning-enabled Relation aggregation (DFLR), based on relation embedding aggregation for auto insurance anomaly detection. Insurance companies participating in this framework will be either in the client role or server role throughout the training. For the client role, it locally executes KGE training, while the server performs average aggregation on relation embeddings to enhance the expression capability of embeddings.
- We design a dynamic server selection mechanism that continuously reorganizes federation groups based on data dissimilarity among clients. This approach elevates training efficiency and quality while ensuring privacy protection.
- We conduct experiments on a real auto insurance dataset. The effectiveness of DFLR is successfully demonstrated through comparative experiments. In particular, the ComplEX+SVM model utilizing DFLR can even improve the average prediction precision to 0.6575, which is 0.0898 higher than the result of training only on private data.

The remainder of the full paper is organized as follows: the Section 2 describes related research on DFL and knowledge graph embedding; the Sections 3 and 4 elaborate on the algorithm of the proposed DFLR framework; the Section 5 mainly analyzes the results of comparative experiments; and the Section 5 provides a summary of the full paper as well as an outlook for future research.

2. Related Work

2.1. Decentralized Federated Learning

Artificial Intelligence is steadily developing, and its technical underlying support remains data. The quantity, quality, and dimension of data have become some of the most important factors constraining the progress of science and technology. As data owners need to consider data security protection, competition relations, and legal regulations when facing data exchange and sharing, it leads to the problem of "data silos" between enterprises and industries [21]. How to share data safely and effectively has become a popular research topic.

In 2017, Google first proposed and constructed a Federated Learning (FL) framework to realize the idea of model updating locally [22]. They aimed to improve the prediction accuracy of what Android users associate with their next input when typing on their mobile terminals. Subsequently, a large number of scholars conducted more in-depth research on data security and personalized models. In 2019, Google released the first FL framework in the world, TensorFlow Federated Framework. And in the same year, Professor Yang Qiang with his team open-sourced the first FL framework in China, named Federated AI Technology Enabler [23], as a secure computing framework to support the Federated AI system.

FL can not only break through the "data silos" and "small data" limitations during the training process, but also ensures a certain degree of data privacy and security while benefitting all participants [24]. Because of this, it has been a high concern of researchers in various fields. This framework has a wide range of applications, including medical image processing [25], auto plate recognition [26], air handwriting recognition [27], and so on.

However, due to the high dependence of FL on the central server, it is unable to cope with the problem of a single point of failure of the central server, and thus DFL emerges [28]. A more decentralized federation aggregation is achieved through communication and interaction between participants. Currently, this framework has been applied in several fields. Lu et al. [29] extracted medical patient features more securely by constructing a DFL model that conforms to realistic cooperation. In addition, Kalapaaking et al. [30] combined blockchain with FL to improve the security of the system by using the traceable and untamperable characteristics of blockchain. They used blockchain with a Trusted Execution Environment to replace the central server to improve the fault tolerance and attack resistance of the system.

Compared with traditional FL, DFL has no central communication bottleneck, but it generates a huge client–client communication overhead. To deal with this problem, Liu et al. [31] pioneered the application of the Lloyd–Max algorithm to DFL. They utilized the exchange of model information between neighboring nodes to adaptively adjust the quantization level, and succeeded in improving the communication efficiency by reducing the amount of data of the federated transmission model parameters. Sun et al. [32] investigated Decentralized FedAVG with Momentum (DFedAvgM) based on the FedAVG paradigm, which reduces the communication overhead by mixing the matrices, Momentum, multiple local iterations of client training, and quantization of sending models.

2.2. Knowledge Graph Embedding

Knowledge Graph as a kind of mesh database manages loose multi-source heterogeneous data through a standardized structural organization (head entity, relation, tail entity) [33]. With the advantages of graph structure to reflect and manage information, and to help accurate positioning and searching, KG provides strong underlying support for specific downstream applications such as Internet semantic searches, personalized recommendations, an intelligent Question Answering System and big data decision-making.

However, such a ternary structure is difficult to deal with directly due to the low portability of its underlying symbolic properties. Therefore, researchers apply knowledge graph embedding to ensure computational simplicity by embedding entities and relations into a continuous low-dimensional vector space, while preserving the structural information of the KG [34]. The entities and relations are downscaled and stored in the low-dimensional space in the form of vector matrices or tensors.

The training of KGE involves the semantic understanding level. It is necessary to consider how to extract relations and entities from non-aligned heterogeneous data, and how to understand the real meaning of different relations and entities for the alignment task [35].

Existing knowledge graph embedding methods are mainly categorized into three types: (i) Translational Distance Models; (ii) Semantic Matching Models; and (iii) Neural Network Models. Based on the above three types of models, many models have been derived.

The Translational Distance Models are based on TransE [36] and extend to derive models such as TransR [37], RotatE [38], and HAKE [39]. This type of method defines the scoring function by modeling the relation as the distance from the head entity to the tail entity like the Euclidean distance of TransE and the rotation transformation of RotatE. Semantic Matching Models measure the rationality of triples at the semantic level to construct score functions, mainly including RESCAL (bilinear model) models [40]. Due to the lack of clear Euclidean inner product correspondence in hyperbolic spaces, in order to extend the calculation to hyperbolic spaces, Ivana Balažević et al. first used a combination of a bilinear model and Poincaré ball [41]. The MuRP model proposed by them can outperform Euclidean models on the link prediction task at lower dimensionality. As for the Neural Network Models, they score by embedding the head entity, relation, and tail entity into the neural network. Jiarui Zhang et al. [42] found that data-driven link prediction tasks rely on various labels and only utilize the structural information of the graph. Inspired by knowledge distillation, the DA-GCN proposed by them makes use of logical rules to reduce the dependence of graph neural networks on data and iterative rules to construct graph convolutional networks. The KGE trained by DA-GCN can perform excellently in link prediction tasks.

3. Proposed Approach

Fraud in the insurance industry usually refers to the behavior of the policyholder, the insured, or the beneficiary to obtain insurance benefits by various means of fabrication, fiction, exaggeration, concealment, and so on [43]. Insurance fraud corresponds to the anomaly detection of entity nodes in a static graph [44]. Consider a scenario where a network represents various insurance claims, policyholders, accident vehicles, and insurance companies. Detecting insurance fraud is akin to identifying irregularities in the interactions between nodes. By treating entities as nodes and their relationships as edges in a graph, one can employ anomaly detection techniques to uncover unusual connections or behaviors. In this section, for the auto insurance anomaly detection task, we will detail the overall framework of the proposed DFLR and introduce the method of server aggregation embedding and the process of client updating local embedding, respectively. In addition, this section includes the server selection mechanism we designed in DFLR.

The training process for one round of decentralized federated training is shown in Figure 2. For the purpose of facilitating the subsequent description of the algorithm and framework, we define some relevant terms and letters in Table 1.

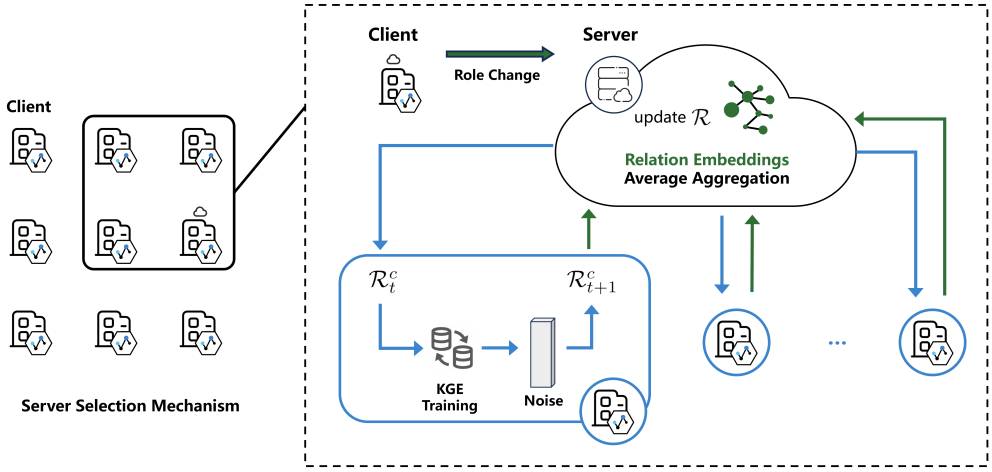

Figure 2. Overview of DFLR.

Table 1. Glossary of notations.

Symbol	Definition
Round t	The complete training process for a full federation round consists of three stages. (i) The server sends the parameters down to the client; (ii) the client receives them for local training; (iii) the client uploads the parameters to the server for aggregation.
N, **K**	The number of clients; the number of the federation group.
E, **B**	The batch size of the client; The training epoch of the client.
$\mathcal{KG}$, $\mathcal{E}$, $\mathcal{R}$	Knowledge Graph set owned by an auto insurance company; the entity embeddings of $\mathcal{KG}$; the relation embeddings of $\mathcal{KG}$.
S_t, S_t^j, C^j	Set of servers at round t; the server of the j-th federation group; the client's set of the j-th federation group.
Z	The fixed number of rounds that the client needs to participate in in federated training before implementing the server selection mechanism.
P	The proportion of clients within the same federation group which choose to regroup.

3.1. DFLR Framework

In the auto insurance industry, it is virtually impossible for car owners to sign up for the same car insurance policy between different insurance companies, subject to state laws and auto insurance claim rules. For example, as a rule, an owner can only carry one Compulsory Third Party Liability Insurance (CTPL) policy on a specific vehicle. Thus, there is less overlap of entities, and cross-company federal aggregation of these entities makes little sense. On the contrary, in each company's private data, the relations between entities are similar, such as "own", "policy holder", and "overhaul at". The most important thing is that these simple relation embeddings hardly involve the details of the owner and vehicle. By sharing these relation embeddings for FL, the risk of a privacy breach is much less. Therefore, as described in Algorithm 1, after locally training and updating the relation and entity embeddings, the client uploads the relation embeddings to the server. As for the server, it is responsible for the aggregation of the relation embeddings in the whole training framework.

In a real enterprise competition scenario, it is quite challenging for all auto insurance companies to agree to work closely together, and the depth of cooperation varies from company to company. Therefore, following the traditional framework of FL, establishing only one central server appears more utopian when multiple companies cooperate. The introduction of the DFL model is realistic and reasonable. In DFL, the relation is more free and flexible compared to the structure under the traditional FL framework. As shown in

Figure 3, any client node in the DFL framework can act as a central server for federation aggregation, so its role can be either client or server.

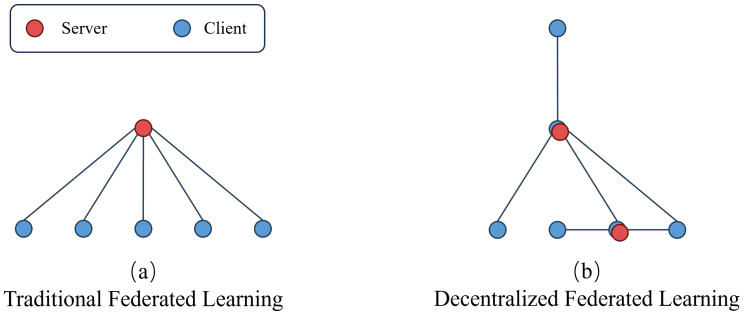

Figure 3. Traditional FL and Decentralized FL framework.

Accordingly, in order to better accomplish the auto insurance anomaly detection task, we focus on the following three issues: (i) the server aggregation algorithm; (ii) the local model of clients; and (iii) the server selection mechanism for each round of DFL.

Algorithm 1 The DFLR framework.

Server Aggregation
Initialize relation embeddings R_0 and select the server set S_0 for the first round of federation.
for round t = 0, 1, 2,... **do**
for $S_t^j \in S_t$ **in parallel do**
Servers S_t^j distributes $\mathcal{R}_t^j$ to clients C^j.
for $C_m^j \in C^j$ **in parallel do**
$\mathcal{R}_m^j \leftarrow$ **Client Training**
end for
Update $\mathcal{R}_{t+1}^j$ based on FedAVG.
end for
end for

Client Training
Receive relation embeddings from server as $\mathcal{R}_t^c$.
for e = 1,..., **E do**
Sample $\mathcal{KG}_{batch} \subseteq \mathcal{KG}$ of size B;
for each $(h, r, t) \in \mathcal{KG}_{batch}$ **do**
Sample a negative triple (h, r, t')
Compute loss $\mathcal{L}$ by Equation (1)
Update embeddings $\nabla\mathcal{L}$
end for
Add Gaussian noise to r
end for
Execute server selection mechanism every **Z** rounds.
return $\mathcal{R}_{t+1}^c$

3.1.1. Server Aggregation

One of the key modules of this DFLR framework, the federated aggregation method, is based on the FedAVG [45] proposed by Mcmahan et al. in 2017. Computing by averaging can help individual clients within a federation group to achieve embedding consistency and cooperation. Under the traditional FL, there are two types of roles: (i) client—the company or institution that jointly trains; and (ii) server—a unique central server built after unified negotiation among all clients. However, in DFLR, the server is not uniquely determined. During the local training phase, the company maintains the client role, and during the federated aggregation phase, some companies transition from the client role to the server role through the server selection mechanism.

Considering that in the KG data of auto insurance, entity carries too much privacy information compared with relation and the private relations of different clients overlap much more, we chose relation embedding as the data transmitted between clients and servers. Through the collection, average aggregation, and distribution of relation embeddings, joint training cooperation between clients is ultimately achieved.

During the initialization phase, the embeddings of all relations are randomly generated based on the relations owned by all clients, ensuring that the same relation between clients remains consistent regarding embedding before training. Then, the first round of server role selection is carried out. This is based on the statistical results of the number of triples $(?, r, ?)$ that exist in a certain type of relation r for each client, calculating cosine similarity as a measure of similarity between client data. The $(?, r, ?)$ represents a set of triples where the relationship is specified, and the head and tail entities can be any entities. By using the K-means classification method, these clients are divided into K groups. Subsequently, the client with the highest number of triples in one group is selected as the server for the initial round of federation, forming a set of servers S. Each server distributes relation embeddings to all clients within its federation group.

The entity embeddings in the KG of the client are randomly generated locally, and the relation embeddings are distributed by the server. The client uses a KGE model to update all triple embeddings. After training with a specific number of epochs on triple embeddings, the client randomly perturbs relation embeddings. After noise processing, the client uploads relation embeddings to the server selected in the initial round.

The server is responsible for receiving the relation embeddings after training from all clients in its federation group. After receiving all, the server performs weighted average aggregation based on FedAVG. After aggregation by the server, relation embeddings are distributed to various clients. Subsequently, the above training steps are repeated until all embedding effects are no longer improved.

The server is not selected in the initial round and remains unchanged. During the federated training process for all clients, the server is regularly replaced through the server selection mechanism, as well as the federation group for each client. The specific process is detailed in Section 3.1.3.

3.1.2. Client Training

The client applies the KGE training model as a way to extract features. We chose seven typical KGE models to train triples (h, r, t). These models are listed in Table 2, along with their score functions. Additionally, we adopt loss function Equation (1) proposed by Zhiqing Sun et al. [38].

$$\mathcal{L} = \sum_{h,r,t \in \mathcal{KG}} \sum_{h,r,t' \in \mathcal{KG}'} [\gamma + f(h,r,t) - f(h,r,t')]_+ \tag{1}$$

where (h, r, t') means the tail entity of true triple is replaced with the other tail $t \in \mathcal{KG}$, γ denotes a margin hyperparameter, and $[x]_+$ denotes the positive part of x. We construct suitable negative samples by randomly replacing tail entities in the same class, like replacing different Policy IDs. The loss function tends to lower the score of a true triple. After KGE training, it helps to widen the score between positive and negative samples, making the embedding quality of positive samples increase.

In the training phase, negative samples, $\mathcal{KG}'$, need to be constructed by randomly replacing the tail entity. However, the negative triples generated using the randomized method are of poor quality, do not help in training the true triples, and even slow down the convergence. In order to effectively widen the score gap between positive and negative triple samples, it is important to strategically conduct negative sampling. The negative sampling we adopt references the method proposed by Yongqi Zhang et al. which utilizes the concept of negative triple confidence to measure the quality of the negative triples [46]. First, negative triples generated by the uniform random method are used as negative sample candidates. And then, the negative triple with the highest confidence is selected for

training based on the ranking of the confidence score results. According to the definition of triple confidence in CKRL, the confidence of a negative triple is calculated as follows:

$$Q(h, r, t') = -f(h, r, t') \tag{2}$$

$$C(h, r, t') = \frac{\exp Q(h, r, t')}{\sum_{h,r,t' \in \mathcal{KG}'} \exp Q(h, r, t')} \tag{3}$$

where the Q value is based on the score function of the KGE model.

When the client receives the relation embedding transmitted from the server, it re-updates the local triple embedding based on the above embedding training model.

After local training on the client, the relation embeddings are randomly perturbed by adding Gaussian distribution noise to further ensure data privacy. By introducing Gaussian noise, the received embeddings at the server become more blurred and uncertain. This enhances the achievement of data de-identification, preventing attackers from using statistical information to back-propagate training data. Its probability density function is defined as

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x-\mu^2}{2\sigma^2}} \tag{4}$$

where x is a random variable, μ is the mean value, and σ is the standard deviation. Afterwards, the client uploads the noised relation embeddings to the server.

Table 2. KGE training model.

Model	Score Function
TransE [36]: TransE assumes that relations can be represented as translations between entities. It learns embeddings by minimizing the translation distance between the head and tail entities.	$f = \|h + r - t\|_{L_1/L_2}$
TransH [47]: TransH builds on TransE by introducing the concept of hyperplanes. Each relationship has a corresponding hyperplane, and entity embeddings are projected onto these hyperplanes for more flexible modeling.	$f = \|(h - w_r^\top h w_r) + d_r - (t - w_r^\top t w_r)\|_{L_2}^2$
TransF [48]: TransF is an improvement over TransH, using the Frobenius norm to measure the match between entities and relationships.	$f = (h + t)\top t$
RotatE [38]: RotatE employs rotation operations and represents relationships as complex numbers. It models the interaction between entities by rotating them in the complex plane.	$f = \|h \circ r - t\|$
DISTMULT [49]: DISTMULT measures the match between entities and relationships by taking the dot product of their embeddings.	$f = h^\top diag(r) t$
HolE [50]: HolE uses tensor representations to map entities and relationships to a high-dimensional space. It calculates match scores through convolution operations.	$f = r^\top (h * t)$
ComplEx [51]: ComplEx represents entities and relationships using complex numbers. It calculates match scores through complex multiplication, allowing for more flexible modeling of interactions.	$f = Re(h^\top diag(r) \bar{t})$

3.1.3. Server Selection Mechanism

How to select a client to transit from the client role to server roles in a DFL framework is a key issue.

During the initialization phase, as mentioned in the Server Aggregation section, the first round of federated server selection is based on the number of triples in the client's private KG. In the following training phase, the server selection mechanism will automatically proceed after **Z** rounds of federation, and the specific process is as follows:

(1) Each client will record the triple embeddings of the previous **Z** rounds of federation for subsequent traceability and comparison.
(2) If a client's highest F1-score of fraud prediction on the validation set in the latest **Z** rounds is lower than the highest F1-score in the previous **Z** rounds, it means that this client did not benefit from this federation group. When more than **P** percentage of clients within one federation group have not benefited from training, then all clients within that group will be added into the Group Re-selected Client Sequences.
(3.a) For all clients in the Group Re-selected Client Sequences, the client with the highest computing power will calculate the Euclidean distance between the highest quality relation embeddings in the previous **Z** rounds for K-means grouping. After constructing federation groups, the clients within the group are selected again based on the number of triples for the next round of federated training.
(3.b) If client training stagnation occurs only within a single federation group, each client in it will be assigned to other groups. The Euclidean distance is also used as the grouping basis. What is calculated here is the Euclidean distance between the relation embeddings of clients in the Group Re-selected Client Sequences and the relation embeddings of clients who assume server roles within other groups.

One can perform federation group updates and server selection through the above steps.

4. Experimental Analysis

4.1. Evaluation Metrics and Data

For this binary classification of anomaly detection, the experimental evaluation indicators used are listed in Table 3 according to the confusion matrix. They include the True Positive Case (TP), which indicates that an actual Fraudulent auto insurance claim is judged as Fraudulent by the model; the False Positive case (FP), which indicates that a normal auto insurance claim is judged as Fraudulent by the model; the True negative case (TN), which indicates that a normal auto insurance claim is judged as normal by the model; and the False negative case (FN), which indicates that a Fraudulent auto insurance claim is judged as normal by the model.

The experimental evaluation metrics are selected as the classical indicators for binary classification problems, which are precision, recall, F1-score, and accuracy.

Table 3. Anomaly detection confusion matrix.

Real Value	**Predicted Value**	
	Positive	**Negative**
Positive	TP	FN
Negative	FP	TN

The dataset used in the experiment is from a cooperative auto insurance company. By extracting the entities in auto insurance cases, the graph is organized as shown in Figure 4, which includes five categories, 14 types of entities, and some main relations between entities. Due to the transmission of relations, some relations are not shown in Figure 4. After a series of matching and MD5 encryption operations, preliminary privacy protection was applied to data related to the personal information of the policyholder, plate number, driver's licenses, etc. After a series of data pre-processing, the dataset consisted of 725,388 triples, 11 relations, and 81,381 entities.

For the purpose of simulating the private data differences among clients, we divided the dataset into 6 and 12 clients through relation, and the average number of KGs is shown in Table 4. And we randomly deleted some relations so that the relations between clients are not completely consistent. In addition, in order to ensure the quality of embeddings, we split the dataset into training, validation, and testing sets in an 8:1:1 ratio.

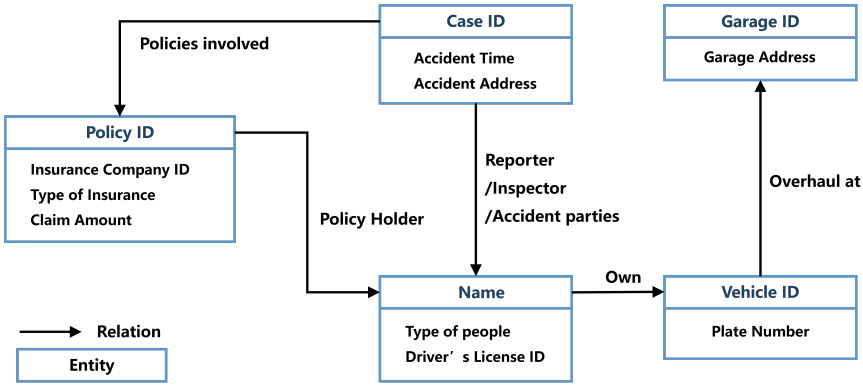

Figure 4. Entity and relation of auto insurance Knowledge Graph.

Table 4. The average number of KGs owned by split clients.

Number of Clients N	Triples	Relations	Entities
6	120,898	9	13,563.5
12	60,449	10.45	671.75

4.2. Implementation

We set up three training modes to verify whether the DFLR method is effective, including Single-Client, All-Clients, and DFLR-Clients. The meanings of these three modes are as follows:

- **Single-Client:** In this mode, there is no cooperation between clients and they are trained using only their own private data.
- **All-Clients:** In this mode, competition between clients is ignored and no privacy protection is adopted. All clients pool their private data together, train a single detection model uniformly, and finally test the model effect with a test dataset on each client.
- **DFLR-Clients:** In this model, each client has the ability to switch roles, including client or server. In the local training phase, all clients only rely on local data for training. In the federation phase, the server selection mechanism is performed first, and some of the client identities are switched to servers. After collecting the relation embeddings uploaded by the clients in the group, the server performs average aggregation based on FedAVG, and then distributes them. Finally each client evaluates the model effect on its own test dataset.

In the experiment, we construct different processes using the same type of GPU devices with the same configurations to simulate the clients for local training. And an SVM anomaly detection model [52] is applied after local KGE training. We set the embedding dimension trained in three modes as 256, the learning rate as 0.001, and the margin γ of Equation (1) as 1.0. For DFLR-Clients mode, the client training batch size **B** was 512, local training epoch **E** was five, and **Z** was four. As for the SVM model, we set the C parameter to 0.01 and the gamma parameter to 0.001.

To assess the training efficiency of the federation rounds, it was set that when the accuracy rate was no longer decreasing within six rounds, the federated training was ended, and the experimental effect was evaluated using the above-mentioned metrics. The accuracy of the validation set was evaluated every 10 epochs for training modes Single-Client and All-Clients, and every 5 Federation rounds for DFLR-Clients. The model fitting effect was ensured by using the early stopping method, and the whole training session ended when the accuracy was no longer improved for 12 consecutive epochs/rounds on the validation set. The trained model was then used on the test set for metrics evaluation.

4.3. Experiments

Our experiment mainly focuses on three important questions: (i) verifying whether the DFLR training mode is effective; (ii) detecting the impact on a specific client as the number of federated participants increases; and (iii) detecting the impact on a specific client as the number of federation groups increases.

4.3.1. Verify the Effectiveness of DFLR Training Mode

In this experiment, we set the number of clients **N** to six and the number of federation groups **K** in DFLR-Clients mode to three. The results of the three modes of training are listed in Tables 5 and 6. The scarcity of anomalous samples in the dataset resulted in relatively low F1-scores in these tables. However, there are still gaps in the predicted results that can be used for analysis under different mode settings.

Table 5 shows the average results for six clients. This table shows that our proposed DFLR successfully improved the average training performance of all clients compared to the Single-Client mode. Especially under the DFLR framework, the **TransH+SVM** and **HolE+SVM** models performed even better than the results of All-Clients mode. And in all DFLR-Clients training modes, the **HolE+SVM** model achieved the highest results, with 0.4822 on precision, 0.3823 on recall, and 0.4265 on F1-score. As mentioned in the above examples, some results show that DFLR-Clients mode can surpass All-Clients mode. This may be because the DFLR-Clients mode enables clients to cooperate with clients with higher embedding similarity. All-Clients, on the other hand, simply centralizes clients' data and does not take into account the differences between clients. In the All-Clients mode, there is a situation that the results of some clients are pulled down, so the average results will be relatively low.

Table 6 is the test result of one of the clients. The detection results of fraud in DFLR-Clients mode show that the embedding quality of the client was also effectively improved through training. The **ComplEx+SVM** model even achieved the optimal effect in DFLR-Clients mode. This result may be due to a significant deviation in the data distribution between Client-1 and other clients in the ALL-Clients mode. And the DFLR-Clients mode utilizes a reasonable grouping mechanism, thereby improving the prediction results of Client-1.

Based on the above results, we can verify the effectiveness of our proposed DFLR framework. The DFLR framework successfully improved the improvement limitation of single client training, achieving the benefits of each participant through federated cooperation. The DFLR framework successfully breaks through the limitations of single client training and benefits each participant through federated cooperation.

Table 5. The average results of 6 clients in 3 different training modes.

Model	Training Mode	The Average Results of 6 Clients		
		Precision	**Recall**	**F1-Score**
TransE + SVM	Single-Client	0.2675	0.2016	0.2299
	All-Clients	0.4283	0.2763	0.3359
	DFLR-Clients	**0.4029**	**0.2372**	**0.2987**
TransH + SVM	Single-Client	0.2862	0.2348	0.2580
	All-Clients	0.4079	0.2783	0.3309
	DFLR-Clients	**0.4183**	**0.2902**	**0.3427**
TransF + SVM	Single-Client	0.2822	0.1928	0.2291
	All-Clients	0.4324	0.2863	0.3445
	DFLR-Clients	**0.3739**	**0.2769**	**0.3182**

Table 5. *Cont.*

Model	Training Mode	The Average Results of 6 Clients		
		Precision	**Recall**	**F1-Score**
RotatE + SVM	Single-Client	0.3001	0.2531	0.2746
	All-Clients	0.4421	0.3142	0.3673
	DFLR-Clients	**0.4267**	**0.2742**	**0.3339**
DISTMULT + SVM	Single-Client	0.3093	0.2361	0.2678
	All-Clients	0.4812	0.4046	0.4396
	DFLR-Clients	**0.4728**	**0.3836**	**0.4236**
HolE + SVM	Single-Client	0.3533	0.2363	0.2832
	All-Clients	0.4529	0.3740	0.4097
	DFLR-Clients	**0.4822**	**0.3823**	**0.4265**
ComplEx + SVM	Single-Client	0.2683	0.1578	0.1987
	All-Clients	0.5026	0.3822	0.4342
	DFLR-Clients	**0.4508**	**0.3878**	**0.4169**

* The bold data in the table indicate that the DFLR-Clients mode outperforms the Single-Client mode, and the underlined data indicate that the DFLR-Clients mode is the best of the three modes.

Table 6. The results of Client-1 in 3 different training modes.

Model	Training Mode	Precision	Recall	F1-Score
TransE + SVM	Single-Client	0.5283	0.1827	0.2715
	All-Clients	0.6382	0.3740	0.4716
	DFLR-Clients	**0.5392**	**0.2216**	**0.3141**
TransH + SVM	Single-Client	0.5503	0.2012	0.2947
	All-Clients	0.6821	0.3872	0.4940
	DFLR-Clients	**0.5927**	**0.2672**	**0.3683**
TransF + SVM	Single-Client	0.4928	0.1822	0.2660
	All-Clients	0.6519	0.2426	0.3536
	DFLR-Clients	**0.6222**	**0.2229**	**0.3282**
RotatE + SVM	Single-Client	0.5113	0.1729	0.2584
	All-Clients	0.6018	0.2537	0.3569
	DFLR-Clients	**0.5762**	**0.2322**	**0.3310**
DISTMULT + SVM	Single-Client	0.4636	0.2273	0.3050
	All-Clients	0.6162	0.3093	0.4119
	DFLR-Clients	**0.5127**	**0.2292**	**0.3168**
HolE + SVM	Single-Client	0.5412	0.2282	0.3210
	All-Clients	0.6296	0.3527	0.4521
	DFLR-Clients	**0.6188**	**0.2928**	**0.3975**
ComplEx + SVM	Single-Client	0.5677	0.3015	0.3938
	All-Clients	0.6321	0.3746	0.4704
	DFLR-Clients	**0.6575**	**0.3772**	**0.4794**

* The bold data in the table indicate that the DFLR-Clients mode outperforms the Single-Client mode, and the underlined data indicate that the DFLR-Clients mode is the best of the three modes.

4.3.2. The Impact of the Number of Participants **N** on One Specific Client

Considering that the premise of FL is that multiple clients choose to cooperate for mutual benefit, we conducted a comparative experiment about the number of clients **N** participating in the cooperation. In this experiment, we used data split into 12 clients and set **K** = 2. For Client-1, we sequentially increased the number of clients participating in the DFLR. The accuracy result of Client-1 is shown in Figure 5. It intuitively reflects that as **N** continues to increase, for a certain participant, the ability to detect auto insurance fraud improved, which means an improvement in the quality of KGE.

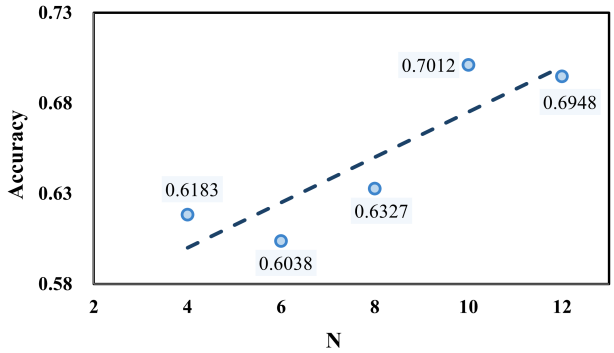

Figure 5. The impact of **N** on clients in DFLR-Clients mode.

4.3.3. The Impact of the Number of Federation Groups **K** on One Specific Client

We used data split into 12 clients and set **K** = 2, 3, 4, 6 so as to observe the impact of **K** on a specific client. Figure 6 shows the F1-score results for two clients. The linear fitting results indicate that increasing the number of federation groups can improve the performance of the client.

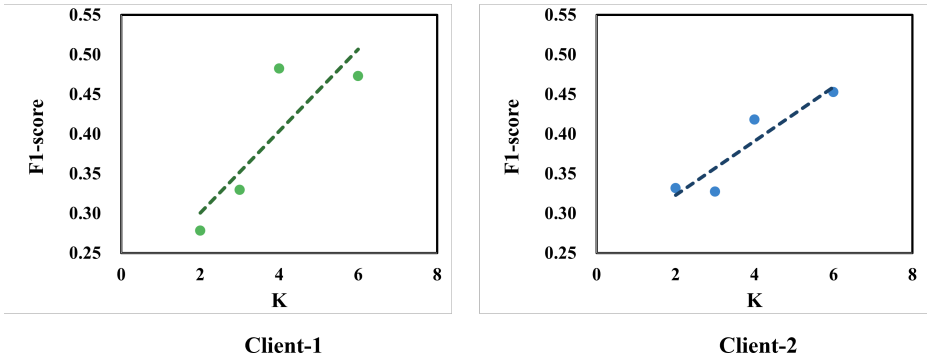

Figure 6. The impact of **K** on clients in DFLR-Clients mode.

5. Conclusions and Discussion

The proposed DFLR framework takes into account the high difficulty of achieving unified cooperation among auto insurance companies in real situations, and adopts dynamic server changes to solve the utopian cooperation of traditional FL. The DFLR uses simple and efficient KGE models for information mining on the local client, and aggregates the actual overlapping relations of auto insurance companies on the server to protect privacy. The experiment has proven that the DFLR framework has higher accuracy than single client training, which is more practical under real-world cooperative relations.

In experiments, we did not fully consider data heterogeneity, but used data with the same structure on the client for federal simulation. In real scenarios., different companies have different requirements for the storage format and structure of data, which largely blocks the algorithm from obtaining qualitative improvement in real-time anomaly detection. Moreover, the data heterogeneity between all the data of different companies, including label offset, data volume imbalance, and so on, was not fully considered. In the future, efforts will be made to improve the efficiency of the data processing process. In addition, considering the reality of the real-time addition of cases, auto insurance detection for static auto insurance data will be expanded to dynamic interaction detection for data updates in the federal round to meet the real-time detection requirements of auto insurance cases. And it is necessary to consider the integrity of cooperation between agencies and companies to prevent the emergence of malicious participants in the federal sharing session. Based on game theory, we could start with incentives for cooperation to stimulate cooperative firms and agencies to reduce destructive behavior in future work.

Author Contributions: Conceptualization, S.S. and X.K.; methodology, S.S.; software, S.S. and H.B.L.; validation, S.S., Z.H., and H.B.L.; investigation, Z.H. and S.S.; resources, X.K. and B.Z.; data curation, Z.H. and B.Z.; writing—original draft preparation, S.S., Z.H., and B.Z.; writing—review and editing, X.K., H.B.L., and S.S.; supervision, B.Z. and X.K.; project administration, S.S. and X.K.; funding acquisition, X.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the National Natural Science Foundation of China (62072409) and the Zhejiang Provincial Natural Science Foundation (LR21F020003).

Data Availability Statement: The auto insurance data used in our experiment were provided by a real cooperative insurance company. Due to the confidentiality agreement, the data cannot be shared.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, H.; Fu, T.; Du, Y.; Gao, W.; Huang, K.; Liu, Z.; Chandak, P.; Liu, S.; Van Katwyk, P.; Deac, A.; et al. Scientific discovery in the age of artificial intelligence. *Nature* **2023**, *620*, 47–60. [CrossRef]
2. Zhang, L.; Wu, T.; Chen, X.; Lu, B.; Na, C.; Qi, G. Auto Insurance Knowledge Graph Construction and Its Application to Fraud Detection. In Proceedings of the 10th International Joint Conference on Knowledge Graphs, Virtual, 6–8 December 2021; Association for Computing Machinery: New York, NY, USA, 2022; pp. 64–70.
3. Dhieb, N.; Ghazzai, H.; Besbes, H.; Massoud, Y. Extreme Gradient Boosting Machine Learning Algorithm for Safe Auto Insurance Operations. In Proceedings of the 2019 IEEE International Conference on vehicular Electronics and Safety (ICVES), Cairo, Egypt, 4–6 September 2019; pp. 1–5.
4. Zhang, R.; Che, T.; Ghahramani, Z.; Bengio, Y.; Song, Y. MetaGAN: An Adversarial Approach to Few-Shot Learning. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 2365–2374.
5. Vincent, V.; Wannes, M.; Jesse, D. Transfer learning for anomaly detection through localized and unsupervised instance selection. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 6054–6061.
6. Chen, Z.; Duan, J.; Kang, L.; Qiu, G. Supervised Anomaly Detection via Conditional Generative Adversarial Network and Ensemble Active Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 7781–7798. [CrossRef]
7. Li, W.; Du, Q. Collaborative Representation for Hyperspectral Anomaly Detection. *IEEE Trans. Geosci. Remote Sens.* **2014**, *53*, 1463–1474. [CrossRef]
8. Peng, H.; Li, H.; Song, Y.; Zheng, V.; Li, J. Differentially Private Federated Knowledge Graphs Embedding. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Virtual, 1–5 November 2021; pp. 1416–1425.
9. Benedek, B.; Ciumas, C.; Nagy, B.Z. Automobile insurance fraud detection in the age of big data–A systematic and comprehensive literature review. *J. Financ. Regul. Compliance* **2022**, *30*, 503–523. [CrossRef]
10. Bhowmik, R. Detecting Auto Insurance Fraud by Data Mining Techniques. *J. Emerg. Trends Comput. Inf. Sci.* **2011**, *2*, 156–162.
11. *China Life 2022 Annual Property and Casualty Claims Settlement Service Report*; Technical Report; China Life Insurance Company: Beijing, China, 2022.
12. Liang, C.; Liu, Z.; Liu, B.; Zhou, J.; Li, X.; Yang, S.; Qi, Y. Uncovering insurance fraud conspiracy with network learning. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, 21–25 July 2019; pp. 1181–1184.
13. Askari, S.M.S.; Hussain, M.A. IFDTC4.5: Intuitionistic fuzzy logic based decision tree for E -transactional fraud detection. *J. Inf. Secur. Appl.* **2020**, *52*, 102469. [CrossRef]
14. Lin, T.H.; Jiang, J.R. Credit Card Fraud Detection with Autoencoder and Probabilistic Random Forest. *Mathematics* **2021**, *9*, 2683. [CrossRef]
15. Sanober, S.; Alam, I.; Pande, S.; Arslan, F.; Rane, K.P.; Singh, B.K.; Khamparia, A.; Shabaz, M. An Enhanced Secure Deep Learning Algorithm for Fraud Detection in Wireless Communication. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 6079582. [CrossRef]
16. Kim, M.; Choi, J.; Kim, J.; Kim, W.; Baek, Y.; Bang, G.; Son, K.; Ryou, Y.; Kim, K.E. Trustworthy Residual Vehicle Value Prediction for Auto Finance. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 15537–15544.
17. Jin, C.; Wang, J.; Teo, S.G.; Zhang, L.; Chan, C.; Hou, Q.; Aung, K.M.M. Towards end-to-end secure and efficient federated learning for xgboost. In Proceedings of the AAAI International Workshop on Trustable, Verifiable and Auditable Federated Learning, Vancouver, BC, Canada, 2 March 2022.
18. Hilal, W.; Gadsden, S.A.; Yawney, J. Financial Fraud: A Review of Anomaly Detection Techniques and Recent Advances. *Expert Syst. Appl.* **2022**, *193*. [CrossRef]
19. Chen, X.; Jia, S.; Xiang, Y. A Review: Knowledge Reasoning over Knowledge Graph. *Expert Syst. Appl.* **2020**, *141*, 112948. [CrossRef]
20. Qu, Y.; Gao, L.; Luan, T.H.; Xiang, Y.; Yu, S.; Li, B.; Zheng, G. Decentralized Privacy Using Blockchain-enabled Federated Learning in Fog Computing. *IEEE Internet Things J.* **2020**, *7*, 5171–5183. [CrossRef]

21. Kong, X.; Gao, H.; Shen, G.; Duan, G.; Das, S.K. FedVCP: A Federated-Learning-Based Cooperative Positioning Scheme for Social Internet of Vehicles. *IEEE Trans. Comput. Soc. Syst.* **2022**, *9*, 197–206. [CrossRef]
22. Chen, M.; Zhang, W.; Yuan, Z.; Jia, Y.; Chen, H. Fede: Embedding knowledge graphs in federated setting. In Proceedings of the 10th International Joint Conference on Knowledge Graphs, Virtual, 6–8 December 2021; pp. 80–88.
23. Federated AI Technology Enabler. Available online: https://github.com/FederatedAI/FATE (accessed on 21 August 2023).
24. Li, L.; Fan, Y.; Tse, M.; Lin, K.Y. A review of applications in federated learning. *Comput. Ind. Eng.* **2020**, *149*, 106854. [CrossRef]
25. Niu, Y.; Deng, W. Federated learning for face recognition with gradient correction. In Proceedings of the AAAI Conference on Artificial Intelligence, Palo Alto, CA, USA, 22 February–1 March 2022; Volume 36, pp. 1999–2007.
26. Kong, X.; Wang, K.; Hou, M.; Hao, X.; Shen, G.; Chen, X.; Xia, F. A Federated Learning-Based License Plate Recognition Scheme for 5G-Enabled Internet of Vehicles. *IEEE Trans. Ind. Inform.* **2021**, *17*, 8523–8530. [CrossRef]
27. Kong, X.; Zhang, W.; Qu, Y.; Yao, X.; Shen, G. FedAWR: An Interactive Federated Active Learning Framework for Air Writing Recognition. *IEEE Trans. Mob. Comput.* 2023, *in press*. [CrossRef]
28. Hegedűs, I.; Danner, G.; Jelasity, M. Decentralized learning works: An empirical comparison of gossip learning and federated learning. *J. Parallel Distrib. Comput.* **2021**, *148*, 109–124. [CrossRef]
29. Lu, S.; Zhang, Y.; Wang, Y.; Mack, C. Learn electronic health records by fully decentralized federated learning. *arXiv* **2019**, arXiv:1912.01792.
30. Kalapaaking, A.P.; Khalil, I.; Rahman, M.S.; Atiquzzaman, M.; Yi, X.; Almashor, M. Blockchain-based federated learning with secure aggregation in trusted execution environment for internet-of-things. *IEEE Trans. Ind. Inform.* **2022**, *19*, 1703–1714. [CrossRef]
31. Liu, W.; Chen, L.; Chen, Y.; Wang, W. Communication-Efficient Design for Quantized Decentralized Federated Learning. *arXiv* **2023**, arXiv:2303.08423.
32. Sun, T.; Li, D.; Wang, B. Decentralized federated averaging. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 4289–4301. [CrossRef]
33. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Philip, S.Y. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 494–514. [CrossRef] [PubMed]
34. Huang, X.; Zhang, J.; Li, D.; Li, P. Knowledge graph embedding based question answering. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, Melbourne, Australia, 11–15 February 2019; pp. 105–113.
35. Wang, Q.; Mao, Z.; Wang, B.; Guo, L. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 2724–2743. [CrossRef]
36. Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 1–9.
37. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. Learning entity and relation embeddings for knowledge graph completion. In Proceedings of the AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; Volume 29.
38. Sun, Z.; Deng, Z.H.; Nie, J.Y.; Tang, J. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
39. Zhang, Z.; Cai, J.; Zhang, Y.; Wang, J. Learning hierarchy-aware knowledge graph embeddings for link prediction. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 6–9 May 2020; Volume 34, pp. 3065–3072.
40. Nickel, M.; Tresp, V.; Kriegel, H.P. A three-way model for collective learning on multi-relational data. In Proceedings of the ICML, Bellevue, WA, USA, 28 June–2 July 2011; Volume 11, pp. 3104482–3104584.
41. Balazevic, I.; Allen, C.; Hospedales, T. Multi-relational poincaré graph embeddings. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 4463–4473.
42. Zhang, J.; Huang, J.; Gao, J.; Han, R.; Zhou, C. Knowledge graph embedding by logical-default attention graph convolution neural network for link prediction. *Inf. Sci.* **2022**, *593*, 201–215. [CrossRef]
43. Wang, Y.; Xu, W. Leveraging deep learning with LDA-based text analytics to detect automobile insurance fraud. *Decusion Support Syst.* **2018**, *105*, 87–95. [CrossRef]
44. Akoglu, L.; Tong, H.; Koutra, D. Graph Based Anomaly Detection and Description: A Survey. *Data Min. Knowl. Discov.* **2015**, *29*, 626–688. [CrossRef]
45. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B.A.Y. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics. PMLR, Lauderdale, FL, USA, 20–22 April 2017; Volume 54, pp. 1273–1282.
46. Zhang, Y.; Yao, Q.; Shao, Y.; Chen, L. NSCaching: Simple and efficient negative sampling for knowledge graph embedding. In Proceedings of the 2019 IEEE 35th International Conference on Data Engineering (ICDE), Macao, China, 8–11 April 2019; pp. 614–625.
47. Wang, Z.; Zhang, J.; Feng, J.; Chen, Z. Knowledge graph embedding by translating on hyperplanes. In Proceedings of the AAAI Conference on Artificial Intelligence, Quebec City, QC, Canada, 27–31 July 2014; Volume 28.
48. Feng, J.; Huang, M.; Wang, M.; Zhou, M.; Hao, Y.; Zhu, X. Knowledge graph embedding by flexible translation. In Proceedings of the Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning, Cape Town, South Africa, 25–29 April 2016.
49. Yang, B.; Yih, W.; He, X.; Gao, J.; Deng, L. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May, 2015.

50. Nickel, M.; Rosasco, L.; Poggio, T. Holographic embeddings of knowledge graphs. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30.
51. Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; Bouchard, G. Complex embeddings for simple link prediction. *Proc. Mach. Learn. Res.* **2016**, *48*, 2071–2080.
52. Hu, W.; Liao, Y.; Vemuri, V.R. Robust Anomaly Detection Using Support Vector Machines. In Proceedings of the International Conference on Machine Learning, Los Angeles, CA, USA, 23–24 June 2003; pp. 282–289.