electronics

MDPI

Article

Pretrained Language–Knowledge Graph Model Benefits Both Knowledge Graph Completion and Industrial Tasks: Taking the Blast Furnace Ironmaking Process as an Example

Xiaoke Huang and Chunjie Yang *

Control Science and Engineering with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China; xiaokehuang@zju.edu.cn
* Correspondence: cjyang999@zju.edu.cn

Abstract: Industrial knowledge graphs (IKGs) have received widespread attention from researchers in recent years; they are intuitive to humans and can be understood and processed by machines. However, how to update the entity triples in the graph based on the continuous production data to cover as much knowledge as possible, while applying a KG to meet the needs of different industrial tasks, are two difficulties. This paper proposes a two-stage model construction strategy to benefit both knowledge graph completion and industrial tasks. Firstly, this paper summarizes the specific forms of multi-source data in industry and provides processing methods for each type of data. The core is to vectorize the data and align it conceptually, thereby achieving the fusion modeling of multi-source data. Secondly, this paper defines two interrelated subtasks to construct a pretrained language–knowledge graph model based on multi-task learning. At the same time, considering the dynamic characteristics of the production process, a dynamic expert network structure is adopted for different tasks combined with the pretrained model. In the knowledge completion task, the proposed model achieved an accuracy of 91.25%, while in the self-healing control task of a blast furnace, the proposed model reduced the incorrect actions rate to 0 and completed self-healing control for low stockline fault in 278 min. The proposed framework has achieved satisfactory results in experiments, which verifies the effectiveness of introducing knowledge into industry.

Keywords: industrial knowledge graph; blast furnace ironmaking process; multi-task learning; fault diagnosis; self-healing control

Citation: Huang, X.; Yang, C. Pretrained Language–Knowledge Graph Model Benefits Both Knowledge Graph Completion and Industrial Tasks: Taking the Blast Furnace Ironmaking Process as an Example. *Electronics* **2024**, *13*, 845. https://doi.org/10.3390/electronics13050845

Academic Editor: Shen Yin

Received: 15 January 2024
Revised: 20 February 2024
Accepted: 21 February 2024
Published: 22 February 2024

1. Introduction

Industrial knowledge graphs are commonly utilized as an effective method for representing knowledge within industrial processes. Despite their widespread use, there is a limited amount of research dedicated to the application of industrial knowledge graphs (IKGs) in industrial tasks. Instead, the current emphasis is on the extraction of triples from diverse data sources for the purpose of constructing knowledge graphs [1–3].

This is a significant concern within the realm of general knowledge graphs; however, industrial data possess inherent characteristics that set them apart from the data utilized in the construction of general knowledge graphs:

- The abundance of structured data and the limited utilization of unstructured data pose challenges in the extraction and articulation of knowledge.
- The dynamic nature of data necessitates that models possess the capability to effectively process dynamic data.
- Industrial activities cover a broad spectrum, and it is challenging to find a single model that can be universally applied to all tasks. At present, industrial tasks are predominantly tailored to specific models for their respective functions.

To overcome the above challenges and promote the wider use of IKGs, this paper aims to address this issue through two approaches. The first involves maximizing the utilization

of diverse industrial multi-source data to enhance the graph's quality. The second entails constructing fundamental application models grounded in knowledge graphs, thereby enabling knowledge graphs to fulfill a wide range of industrial tasks.

This paper presents a model that is developed based on the operational context of the blast furnace ironmaking process (BFIP). The BFIP is a fundamental process within the iron and steel industry, serving as a crucial step in the conversion of energy and mass flows. The BFIP involves the utilization of coke and flux to reduce iron from iron ore within a high-temperature and high-pressure environment, subsequently directing it to the steelmaking process in the form of liquid pig iron. The BFIP reactor and its associated systems are depicted in Figure 1. The reactants, namely iron ore and coke, are sequentially layered at the top of the furnace through the ore and coke feeding system, resulting in the formation of a lumpy zone that descends gradually due to gravitational forces. Concurrently, high-pressure and high-temperature air, combined with oxygen and pulverized coal processed by the hot blast system, is injected into the reactor via a pipe, creating a tuyere raceway and moving upward. A modern blast furnace can possess a volume of up to 7000 cubic meters, leading to variations in pressure and temperature across different areas of the furnace, thereby influencing diverse reactions. In the cohesive zone, characterized by elevated temperatures, iron oxides are reduced to iron, which is facilitated by the carbon framework formed by the coke to create a dropping zone, ultimately resulting in the collection of liquid iron at the furnace hearth. Periodically, liquid pig iron is discharged from the tap hole and directed to subsequent processing stages [4,5].

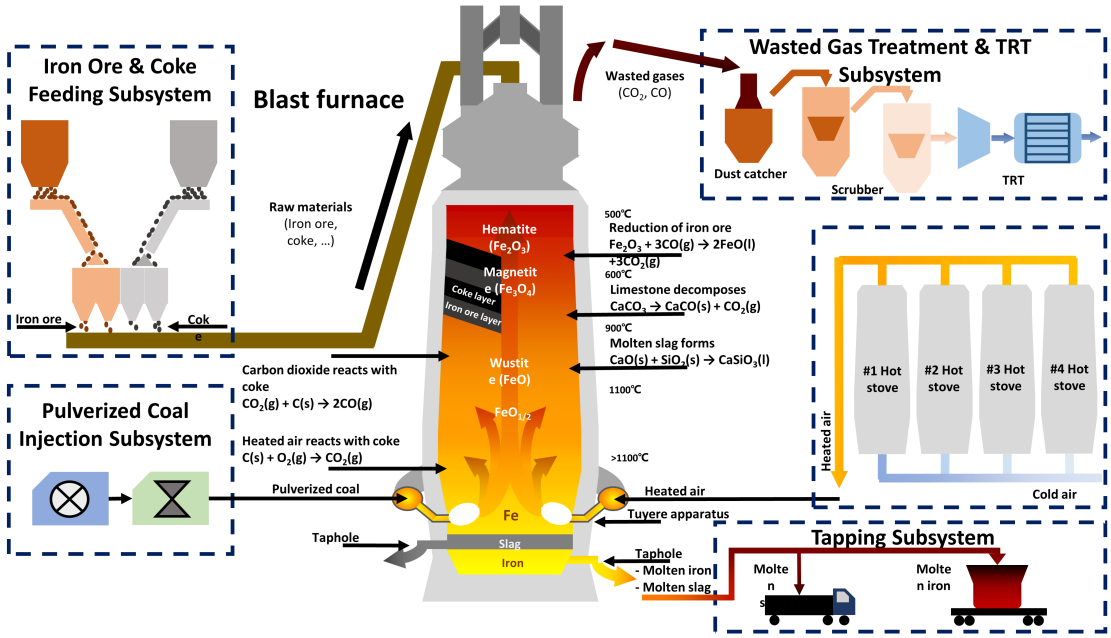

Figure 1. The blast furnace ironmaking process.

In typical furnace operating conditions, the optimization of control variables is essential for enhancing the quality of molten iron or reducing energy consumption, a practice referred to as operation control. Conversely, during abnormal conditions, adjustments to control variables are necessary to rectify faults, a process known as restorative control. A novel control category, termed self-healing control, has emerged, involving the automatic implementation of restorative control using a computer. This approach is gaining attention due to its superior accuracy and response speed compared to manual operation [6].

The faults of the BFIP can be categorized into two groups: (1) single-factor faults, which stem from equipment malfunctions or irregular production scheduling, such as low material line caused by hopper jamming; and (2) systemic faults, which arise from

physical movement or chemical reaction irregularities resulting from mismatches between raw materials and the control system, such as the pipeline caused by abnormal gas flow distribution. Therefore, the goal of self-healing control is to correct the fault and restore the control system to its normal operating state, thereby meeting the strict safety requirements of the BFIP.

Therefore, based on the top-down construction of blast furnace knowledge graphs based on ontology, this article explores the use of production data and unstructured text to complete the graph, further improving the quality of the graph, while constructing a model that can meet different industrial tasks, improving the generality of the model, and reducing the difficulty of completing subsequent industrial tasks based on knowledge graph construction models.

The contributions of this paper are as follows:

- The specific forms of multi-source data in industry are summarized and processing methods for each type of data are provided.
- A pretrained language–knowledge graph model is introduced, which can align graph nodes with text concepts. This model leverages masked entity prediction and link prediction tasks to facilitate the extraction and integration of information from both textual and graphical data components.
- A two-stage model construction strategy is suggested, involving the segmentation of the model into pretraining and fine-tuning stages. The pretraining phase is designed to be applicable to various subsequent tasks, while the fine-tuning phase is optimized using a dynamic expert network structure to enhance its adaptability to industrial process dynamic data and to yield a more precise model for industrial applications.

2. Related Work

2.1. Knowledge Graph Completion

The construction and application of an IKG is a long process, as shown in Figure 2. The process of extracting triples from diverse sources of data, such as text, sensor, and digital twins data, is followed by the alignment of entities, disambiguation, and other related operations. Subsequently, the acquired knowledge can be stored in various database formats and regularly verified to maintain the quality of the graph. The application of an integrated knowledge graph (IKG) is extensive, encompassing areas such as recommendation systems and various other domains [7,8].

In recent years, there has been a growing demand for high-quality knowledge, leading to an increased emphasis on the reliability of information within the context of an integrated knowledge graph (IKG). Consequently, the issue of knowledge graph completion, which involves determining the validity of triples within the knowledge graph, has garnered significant attention. Various studies have been undertaken to explore knowledge graph completion, focusing on methods for modeling the connectivity patterns between entities in the IKG and developing scoring functions to assess the validity of triples [9–11]. However, these approaches primarily rely on the graph structure and relational information within the existing knowledge graph, limiting their predictive capabilities for triples containing less common entities. To address this limitation, a method combined with a language model has been proposed for knowledge graph completion, leveraging entity descriptions and pretrained language models to enhance predictive accuracy [12].

2.2. Knowledge Graph Embedding

There are two approaches to managing IKGs. One method involves converting the graph into an adjacency matrix and employing graph neural networks and other methodologies to analyze the graph structure represented by the matrix. The other feasible and commonly used approach is to utilize knowledge graph embedding (KGE) techniques to train a low-dimensional vector for each entity and relation, while maintaining the inherent structure of the IKG. The resulting vectors obtained through KGE can then be utilized to enhance the representations of entities in the IKG [13–17]. However, it is noted that although

KGE is specifically tailored for tasks related to knowledge graphs, it is not optimized for applications in industrial settings. Therefore, there is a need for the development of more effective KGE methods to improve industrial tasks.

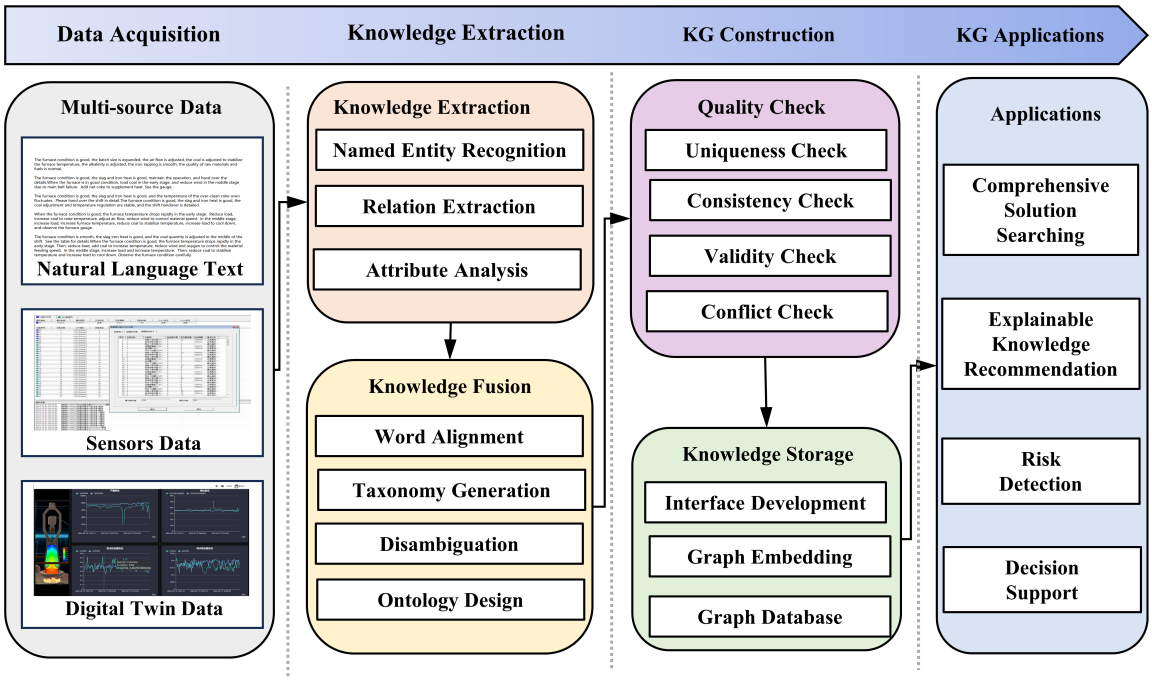

Figure 2. The construction and applications of IKG.

In light of the connection between graph completion and related tasks, some scholars have explored the application of multi-task learning (MTL) strategies to mitigate the impact of disparities between KGE and recommendation tasks, and to enhance the overall performance of both tasks. MTL involves the collective acquisition of knowledge and simultaneous or sequential learning of multiple tasks, which is deemed more advantageous than training them in isolation [18]. In the context of MTL-based recommendation systems, the establishment of links between multiple tasks and the training of these tasks are recognized as pivotal concerns. Recently, a novel MTL approach for knowledge graph enhancement recommendation, known as MKR, has been introduced, demonstrating commendable performance and efficiency in facilitating collaborative enhancement between the KGE task and the recommendation task [19]. Diverging from other MTL-based knowledge graph recommendation systems (KGRSs) that directly utilize vectors derived from KGE to represent users or items in the recommendation system, or simply employ a summation operator to establish connections between the two tasks, MKR achieves interaction between items in the recommendation system and entities in KGE through the utilization of a soft parameter sharing mechanism. MKR employs an alternating training approach within the MTL framework to achieve satisfactory performance, thereby underscoring the efficacy of the parameter sharing mechanism.

2.3. The Application of IKGs in the BFIP

The BFIP, as a conventional industrial process on a large scale, encounters challenges related to standardizing knowledge, insufficient fault data resulting in subpar data-driven modeling, and stringent production safety standards. This presents an opportune situation for implementing an IKG to facilitate knowledge standardization and utilization.

Numerous research studies have been carried out to develop ontological models for the steel industry. These inquiries cover a range of topics including knowledge modeling and information management in the steelmaking process [20], planning and scheduling in primary steel production [21], scheduling and control decision frameworks for process

industries [22], and supply chain decision support for steel manufacturers [23]. However, the focus of the majority of these studies is primarily on supply chain and scheduling issues, with limited attention given to normal production tasks.

This paper aims to improve the knowledge quality of IKGs through multi-source data as well as to complete the production tasks in the BFIP.

3. Methodology

3.1. Industrial Data Collation

Industrial data are commonly sourced from multiple channels and comprise structured, semi-structured, and unstructured data. Structured data, such as sensor-collected variable data stored in databases, are organized in tabular form. Semi-structured data encompass web page content and knowledge graphs containing textual information, while unstructured data include production logs and expert knowledge documents. It is essential to effectively combine and incorporate multi-source data into BFIP modeling in order to enhance the precision and dependability of the model.

Taking the BFIP as an example, the data involved in daily production of the blast furnace include the following categories:

- Structured data: This refers to a tabular dataset housed within a database, comprising information collected from diverse sensors. This typically encompasses process variable data generated during production, such as blast furnace top pressure, hot air pressure, cold air flow, furnace wall temperature, and top temperature. These data are consistently refreshed at a regular sampling rate and centrally archived within the database.
- Semi-structured data: This refers to a localized knowledge graph that encompasses various types of nodes and their interconnections. The information within the graph is typically categorized into three main groups. Firstly, topological data are represented through graph structures and can be articulated in the form of adjacency matrices. Secondly, semantic data are conveyed through heterogeneous triples, which primarily define diverse types of nodes and relationships to facilitate the association of triples for expressing information and knowledge. Lastly, dynamic node information data are preserved through node attributes, which delineate the state of a node within the graph. For instance, the node attribute of a process variable node denotes the value of the node variable at a specific time, while the node attribute of a valve node signifies the opening of the valve.
- Unstructured data: This refers to the maintenance of updated blast furnace logs and expert knowledge documents in the context of blast furnace production. The production process involves a shift system for overseeing and managing production activities, with on-duty conditions being documented in textual format. This includes recording the smelting and tapping conditions of the blast furnace, as well as any adjustments made to the feeding and blowing systems. Consequently, blast furnace logs serve as crucial resources for reflecting the production status of the blast furnace and documenting manual operations. Meanwhile, expert knowledge documents serve as comprehensive compilations of the principles governing blast furnace smelting and control, encompassing the abstraction, summarization, analysis, and generalization of production laws related to blast furnaces.

The three aforementioned data types undergo dynamic updates, although the frequency and methodology of these updates may differ. Additionally, the format and storage approach for each type of data exhibit significant differences. To integrate and model the diverse, multi-source heterogeneous data within a unified framework, it is imperative to preprocess the distinct data types individually. The fundamental concept involves the vectorization of all forms of information and their fusion into vector format, subsequently transmitting them to the network model for processing to achieve the fusion modeling of multi-source information. Given that structured data can be viewed as node attributes of knowledge graphs, it is plausible to combine structured data and knowledge graphs for

processing. Consequently, this paper outlines the processing procedure for semi-structured and unstructured data.

3.1.1. Vectorization of an IKG

Through the utilization of methods for constructing and instantiating ontology, it is feasible to create a comprehensive knowledge graph containing substantial semantic and structural data. One method, referred to as graph embedding, involves transforming the nodes and connections within the graph into a vector space. This procedure maintains the connections between nodes in the vector space and produces vector representations of the nodes. This adaptable representation technique is suitable for a range of applications, making it a widely embraced technology for processing graphs.

The challenges are compounded when a knowledge graph exhibits both dynamic and heterogeneous characteristics. A formal description of a dynamic heterogeneous knowledge graph is provided below.

A dynamic heterogeneous knowledge graph, denoted as $G = (V(t), E(t), A, R)$, consists of an object set $V(t)$ and a link set $E(t)$. A dynamic heterogeneous knowledge graph is also associated with a node type mapping function $f1 : V(t) \rightarrow A$ and a link type mapping function $f2 : E(t) \rightarrow R$. A and R denote the sets of predefined object types and link types, where $A + R > 2$.

Sampling nodes and capturing meaningful changes in the graph structure presents a significant challenge. Recent research has focused on addressing this challenge by exploring dynamic heterogeneous graphs, which integrate previous node embedding techniques for heterogeneous graphs and dynamic graphs to accommodate the evolving nature of heterogeneous graphs [13–16].

3.1.2. Vectorization of Unstructured Data

For unstructured data, in order to obtain word vectors and sentence vectors, we use commonly used text processing processes shown in Figure 3.

- Step 1: This involves preprocessing the text to remove spaces and punctuation.
- Step 2: This focuses on dividing words into sentences and adding a specialized noun dictionary to the BFIP to improve the accuracy of word segmentation.
- Step 3: This pertains to named entity recognition, where a matching template is devised to extract graph nodes from sentences, facilitating effective correspondences between nodes and text within the graph.
- Step 4: This includes utilizing Gensim to train a word vector library suitable for industrial localization.

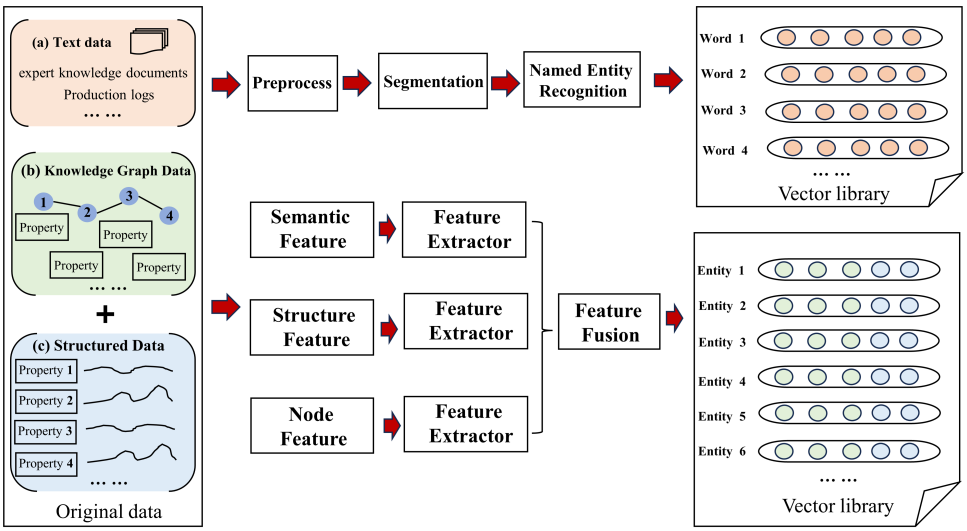

Figure 3. Industrial data collation.

3.2. Two-Stage Model Construction Strategy

Various types of data in industry are stored and updated at different frequencies. For instance, structured sensor data are updated every few seconds or minutes, while unstructured production log data are updated every few hours, and expert documents are updated every few days or longer. This discrepancy in update frequencies presents challenges in information fusion and modeling. A semi-structured knowledge graph serves as both a static knowledge base and a dynamic graph, with the attributes and relationships of nodes constantly changing based on the type of node.

The difference in data update frequency can bring difficulties in information fusion and modeling. In fact, there is a difference in the amount of information brought by data with different frequencies. Data with slower update frequencies contain more information than data with faster update frequencies, especially in describing abstract patterns. To address the technical challenges of multi-scale information fusion and improve information extraction for modeling, this paper proposes a two-stage model construction strategy, as illustrated in Figure 4.

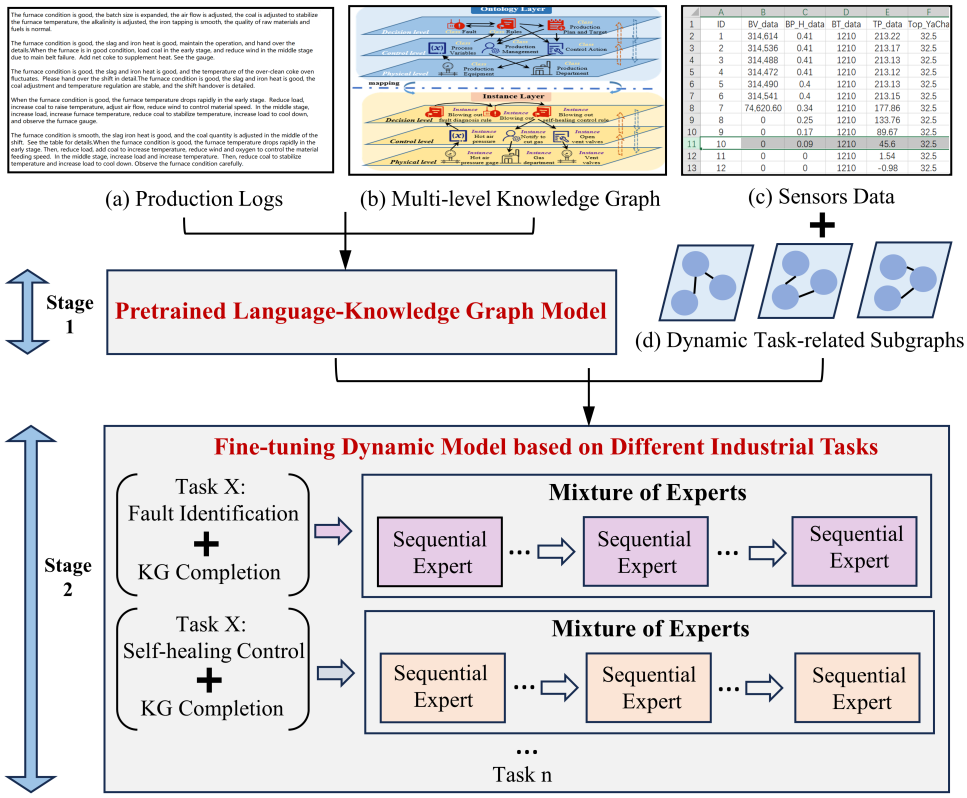

Figure 4. Workflow of the two-stage model construction strategy.

The first stage is to use text data and a static IKG as a knowledge base to construct pretrained models. The model is constructed within a multi-task learning framework, which facilitates feature fusion and sharing at the lower level, and concurrently trains interconnected tasks to acquire feature vectors that amalgamate diverse information. By aligning the concepts in the production log with the nodes in the IKG, the word vectors derived from contextual information in the text are combined with the node vectors obtained from the graph structure. This process yields a pretrained language–knowledge graph model capable of generating features containing comprehensive information. Subsequently, the construction of the second stage for specific dynamic industrial tasks is predicated on this pretrained model.

The second stage is aimed at different dynamic industrial tasks. Based on the vector features provided by the pretrained model, the dynamic data collected from sensors are

further incorporated. These are combined with task-specific dynamic subgraphs sampled from the knowledge graph to facilitate training for particular industrial tasks. Using the MTL framework, two objectives are established: one involves IKG completion, and the other involves industrial tasks such as fault diagnosis, self-healing control, root cause analysis, and other tasks. The feature vectors obtained from the pretrained model in the first stage are utilized, alongside a dynamic graph feature extractor to generate a vector that integrates sensor data and dynamic subgraph features. These two vectors are merged and dynamically inputted into the temporal expert sequence for processing, resulting in the development of a dynamic model capable of simultaneously completing graph tasks and addressing specific industrial challenges.

In the subsequent section, we will furnish a comprehensive elucidation of the two stages.

- We define the production logs set $\mathcal{P}$ as a set of text segments $\mathcal{P} = \{P\}$, and each text segment P as a sequence of tokens (words), $P = (P_1, \ldots, P_i)$.
- We define an IKG as a multi-relational graph $\mathcal{G} = (\mathcal{N}, \mathcal{R})$, where $\mathcal{N}$ is the set of nodes in the IKG and $\mathcal{R}$ is the set of relations. $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{R} \times \mathcal{N}$ is the set of triples that connect nodes in N.
- As the IKG is often large, a dynamic subgraph for specific industrial tasks of the IKG is considered: $G_{sub}(T) = (N, R, T)$ where $N = \{n_1, \ldots, n_j\} \subseteq \mathcal{N}$ and $R \subseteq \mathcal{R}$, the set of triples with time is $E(T) \subseteq N \times R \times N \times T$.
- For node n_k in the IKG, we define the node attribute as A^{n_k}.

3.2.1. Pretrained Language–Knowledge Graph Model

In the context of this study, input instances for the model are generated by pairing a production logs $\mathcal{P}$ with a large IKG $\mathcal{G}$. The objective is to ensure semantic alignment between the text and knowledge graph within each pair, enabling mutual information exchange between the production logs and $G_{sub}(T = 0)$ to facilitate the model's ability to learn interactive reasoning across both modalities. The pretrained model is constructed using the DRAGON framework [24].

The input of the pretrained model is $Input$ = (production logs P, local KG $G_{sub}(T = 0)$). Initially, the input log text is processed using M layers of language model layers to convert it into initial token representations. Additionally, the input nodes of the graph $\mathcal{G}$ are mapped into initial node representations using node embeddings. The embeddings can be denoted as follows:

$$\left(\mathbf{P}_{\text{cross}}^{(0)}, \mathbf{P}_1^{(0)}, \ldots, \mathbf{P}_X^{(0)}\right) = \text{Preprocess} - \text{Layers}(P_{\text{cross}}, P_1, \ldots, P_x) \tag{1}$$

$$\left(\mathbf{N}_{\text{cross}}^{(0)}, \mathbf{N}_1^{(0)}, \ldots, \mathbf{N}_Y^{(0)}\right) = \text{Node} - \text{Embedding}\left(N_{\text{cross}}, N_1, \ldots, N_y\right) \tag{2}$$

In order to capture the reciprocal relationships between the text and IKG, a bidirectional sequence-graph encoder f_{Enconder} is employed. This encoder processes the text tokens and KG nodes, facilitating the exchange of information between them across multiple layers to generate a unified representation for each token and node. Specifically, the model utilizes K layers of fusion layers to collectively encode these representations into the ultimate representations, denoted as:

$$(\mathbf{P}_{\text{cross}}, \ldots, \mathbf{P}_X), (\mathbf{N}_{\text{cross}}, \ldots, \mathbf{N}_Y) = \text{Fusion} - \text{Layers}\left(\left(\mathbf{P}_{\text{cross}}^{(0)}, \ldots, \mathbf{P}_X^{(0)}\right), \left(\mathbf{N}_{\text{cross}}^{(0)}, \ldots, \mathbf{N}_Y^{(0)}\right)\right) \tag{3}$$

where for each of the fusion layers ($\ell = 1, \ldots, K$), the following operation is carried out:

$$\left(\widetilde{\mathbf{P}}_{\text{cross}}^{(\ell)}, \mathbf{P}_1^{(\ell)}, \ldots, \mathbf{P}_X^{(\ell)}\right) = \text{Preprocess} - \text{Layers}\left(\mathbf{P}_{\text{cross}}^{(\ell-1)}, \mathbf{P}_1^{(\ell-1)}, \ldots, \mathbf{P}_X^{(\ell-1)}\right) \tag{4}$$

$$\left(\widetilde{\mathbf{N}}_{\text{cross}}^{(\ell)}, \mathbf{N}_1^{(\ell)}, \ldots, \mathbf{N}_Y^{(\ell)}\right) = \text{GNN} - \text{Layer}\left(\mathbf{N}_{\text{cross}}^{(\ell-1)}, \mathbf{N}_1^{(\ell-1)}, \ldots, \mathbf{N}_Y^{(\ell-1)}\right) \tag{5}$$

$$\left[\mathbf{P}_{\text{cross}}^{(\ell)}; \mathbf{N}_{\text{cross}}^{(\ell)}\right] = \text{Fusion}\left(\left[\widetilde{\mathbf{P}}_{\text{cross}}^{(\ell)}; \widetilde{\mathbf{N}}_{\text{cross}}^{(\ell)}\right]\right) \tag{6}$$

We define a language–knowledge model as a combination of two functions, denoted as $f_{\text{Self}}(f_{\text{Encoder}}(X))$. Here, the encoder function f_{Encoder} takes an input $input =$ (production logs P, local KG $G_{sub}(T = 0)$) and generates contextualized vector representations for each text token, $(\mathbf{P}_1, \ldots, \mathbf{P}_X)$, and for each IKG node, $(\mathbf{N}_1, \ldots, \mathbf{N}_Y)$. The function f_{Self} utilizes these representations to carry out self-supervised tasks. Although any deep bidirectional sequence-graph encoder can be employed for f_{Encoder}, we opt for GreaseLM for the purpose of conducting a controlled comparison with existing studies [25].

In practical terms, in order to carry out the masked language modeling task, the final token in the input text, denoted as $p_x \in P$, is replaced with a special token [MASK]. Subsequently, the task function f_{Self} is implemented as a linear layer that utilizes the contextualized token vectors $\mathbf{P}_x$ from the encoder to predict the original tokens. The objective is to minimize the cross-entropy loss.

$$\mathcal{L}_{\text{Masked}} = -\log proba(p_x \mid \mathbf{P}_X) \tag{7}$$

In this paper, the variables $\textbf{Head} = \mathbf{N}_h$ and $\textbf{Tail} = \mathbf{N}_t$ are defined, along with $\textbf{Relation} = \mathbf{R}_r$, where $\{\mathbf{N}_Y\}$ denotes the contextualized node vectors from f_{Encoder}, and $\mathbf{R} = \left\{\mathbf{r}_1, \ldots, \mathbf{r}_{|\mathcal{R}|}\right\}$ represents the learnable relation embeddings. We analyze an IKG triple scoring function $\psi_r(\textbf{Head}, \textbf{Tail})$. A higher value of ψ indicates a greater likelihood of $(Head, Relation, Tail)$ being a positive triple (edge) rather than a negative one (no edge). Throughout the training process, the objective is to optimize the following:

$$\mathcal{L}_{\text{Linked}} = \sum_{(Head, Relation, Tail) \in E_{out}} \left(-\log \sigma(\psi_r(\textbf{Head}, \textbf{Tail}) + \gamma) + \frac{1}{w} \sum_{(Head', Relation, Tail')} \log \sigma\left(\psi_r\left(\textbf{Head}', \textbf{Tail}'\right) + \gamma\right) \right) \tag{8}$$

where the triplet $(Head, Relation, Tail)$ is associated with w negative samples denoted as $(Head', Relation, Tail')$. The parameter γ represents the margin, while σ denotes the sigmoid function. The underlying rationale of this objective is to encourage the model to classify triplets from the held-out edges E_{out} as positive, while categorizing other randomly selected triplets as negative.

In order to train the model, we concurrently optimize the aforementioned objectives, denoted as $\mathcal{L} = \mathcal{L}_{\text{Masked}} + \mathcal{L}_{\text{Linked}}$. This combined objective integrates the impacts of masked language modeling and IKG link prediction, which promote the model's ability to both align text data with IKG structure and contextualize the IKG with production logs. This facilitates a two-way exchange of information between text data and IKGs for reasoning purposes. The proposed framework is shown in Figure 5.

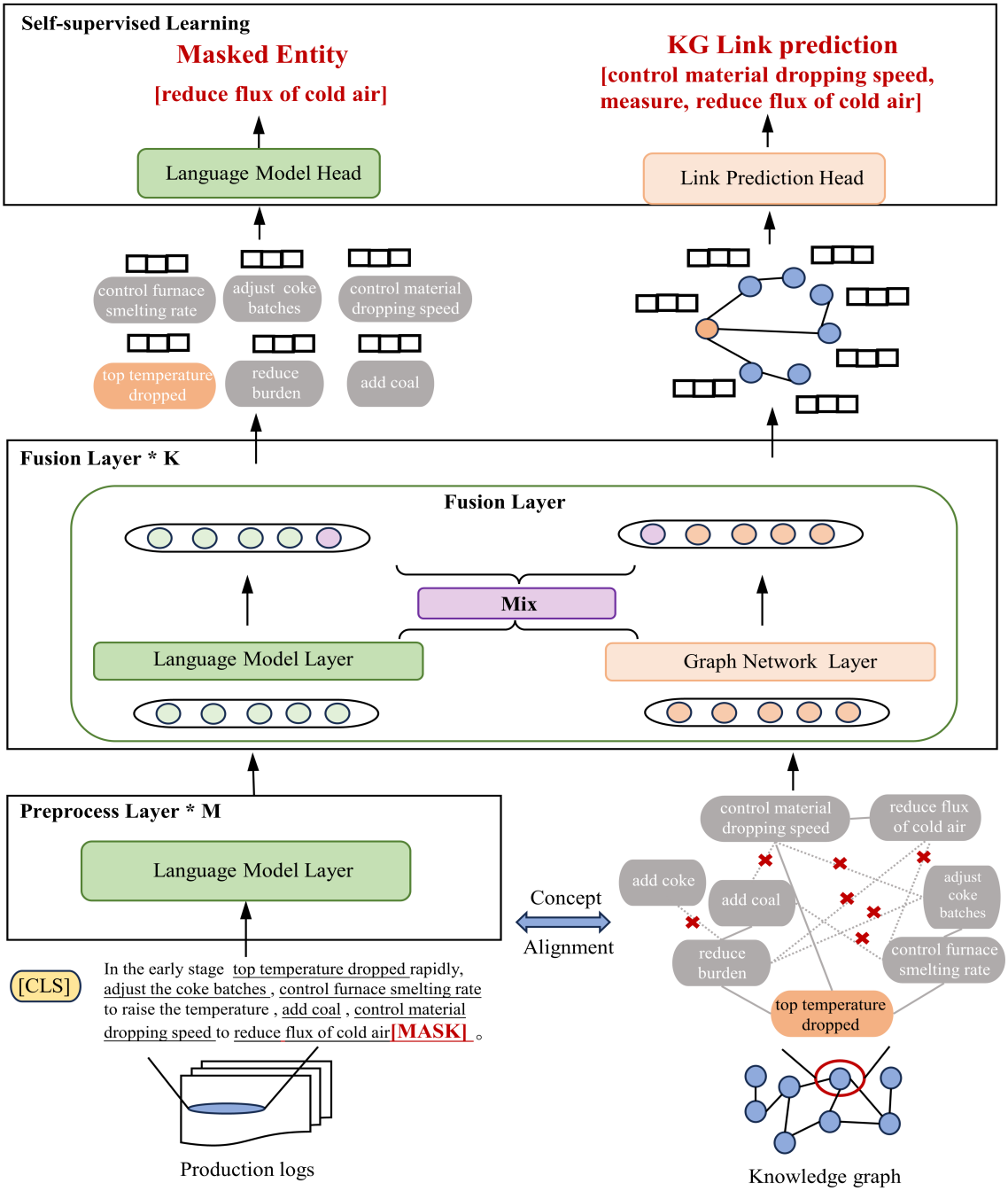

Figure 5. Pretrained language–knowledge graph model.

3.2.2. Dynamic Fine-Tuning Model Based on Industrial Tasks

To enhance the adaptability of the model in the first stage for dynamic industrial tasks, we incorporate an expert network structure to refine the pretrained model. This structure comprises a combination of experts and gates. We extract a pertinent local $G_{sub}(T) = (N, R, T)$ at time T from $\mathcal{G}$. The model takes as input a sequence of $FeaturePair_{(T)} =$ (Text feature P, Subgraph$G_{sub}(T)$, Node featureA^N).

Next, we employ the pretrained model to generate the semantic feature of the text and the topological feature of the IKG. Subsequently, we utilize a dynamic graph feature extractor to produce the integrated topological feature and node feature, denoted as:

$$\left(\mathbf{P}_1^T, \ldots, \mathbf{P}_X^T\right) = \text{Pretrained} - \text{Model}\left(P_1^T, \ldots, P_x^T\right) \tag{9}$$

$$\left(\mathbf{N}_1^T, \ldots, \mathbf{N}_Y^T\right) = \text{Pretrained} - \text{Model}\left(N_1^T, \ldots, N_y^T\right) \tag{10}$$

$$\left(\mathbf{D}_1^T, \ldots, \mathbf{D}_Y^T\right) = \text{GraphFeature} - \text{Extractor}\left(N_1^T, \ldots, N_y^T, A_1^T, \ldots, A_y^T\right) \tag{11}$$

$$\left(\mathbf{F}_1^T, \ldots, \mathbf{F}_Y^T\right) = \text{Concatenate}\left(\mathbf{P}_1^T, \ldots, \mathbf{P}_X^T, \mathbf{N}_1^T, \ldots, \mathbf{N}_Y^T, \mathbf{D}_1^T, \ldots, \mathbf{D}_Y^T\right) \tag{12}$$

The resulting concatenated feature $\mathbf{F}^T = \left(\mathbf{F}_1^T, \ldots, \mathbf{F}_Y^T\right)$ serves as the input for the mixture of experts. This layer consists of a sequence of experts, each specializing in different aspects of the task. Gate networks are employed to control the outputs of the experts, enabling each gate to learn to "select" a subset of experts based on the input example. This mechanism allows for the modeling of intricate interactions among diverse variables. In mathematical terms, a data sequence with s time steps is denoted as:

$$S = \left\{FeaturePair_{(1)}, FeaturePair_{(2)}, \ldots, FeaturePair_{(s)}\right\} \tag{13}$$

We engage in many-to-many sequence learning, as illustrated in Figure 6.

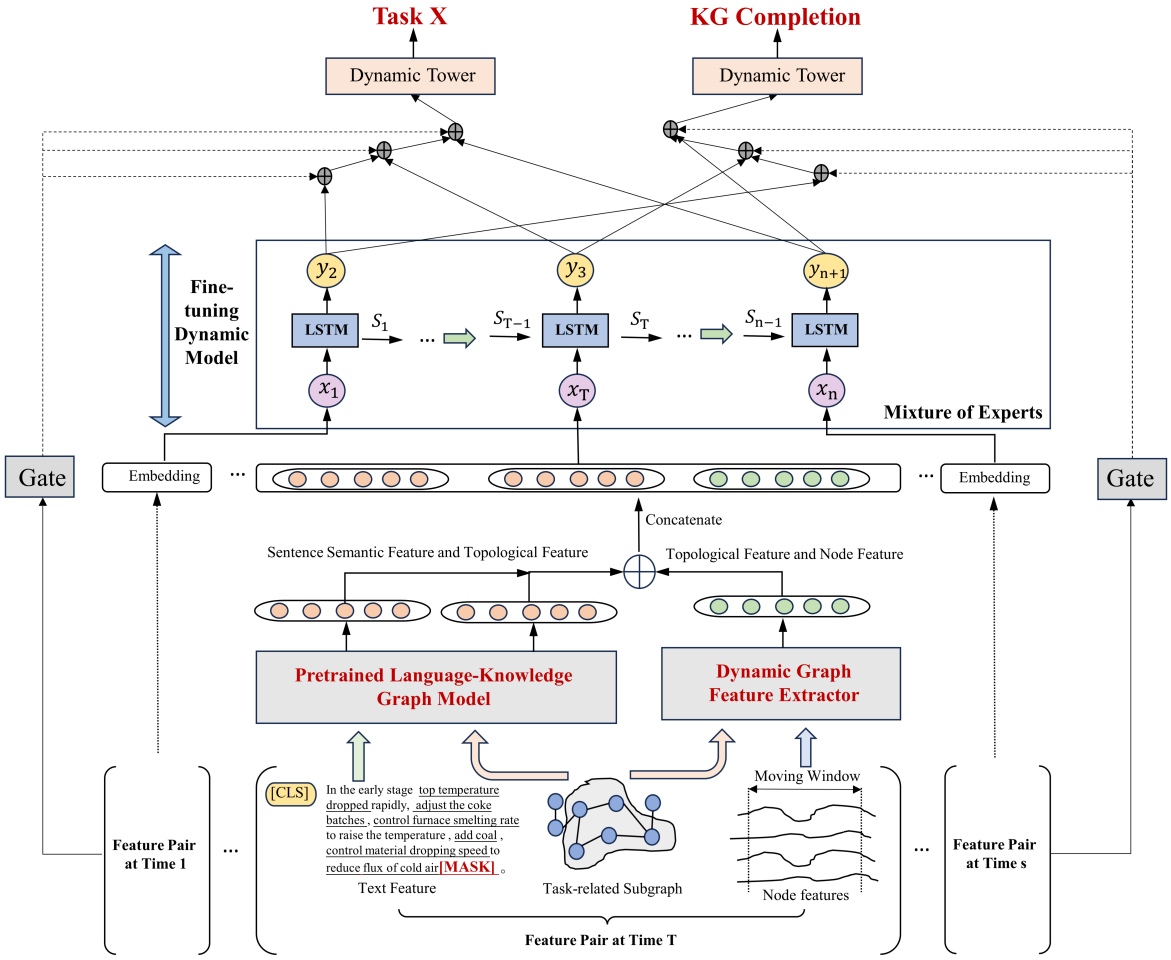

Figure 6. Dynamic fine-tuning model based on industrial tasks .

Given the input data sequence S, we can express the output for task v at time $s+1$ in the following manner:

$$Output_{(s+1)}^v = Tower_{\text{LSTM}}^v\left(Expert^v\left(\mathbf{F^S}\right)\right) \tag{14}$$

$$\text{where } Expert^{v}\left(\mathbf{F^S}\right) = \sum_{i=1}^{z} g^{v}(x)_i Expert_{\text{LSTM}_i}\left(Expert_{\text{LSTM}}\left(Expert\left(\mathbf{F^S}\right)\right)\right) \tag{15}$$

The network denoted as $Tower^{v}_{LSTM}$ is responsible for handling the task labeled as v, $Expert^{v}$ represents the result of the gated mixture of experts layer, while $Expert_{LSTM_i}$ refers to the i-th sequential expert, with a total of z experts. The input to the network is denoted as $\mathbf{F^S}$, and $Tower^{v}$ is the gating network that converts the input into a distribution across the z experts, based on the input:

$$Tower^{v}(\mathbf{F^S}) = \text{softmax}\left(W^{v}_{Tower}\mathbf{F^S}\right) \tag{16}$$

where W^{v}_{Tower} is the weight matrix needing to optimize.

4. Experimental Results and Analysis

4.1. Experimental Setting and Data

This paper uses the pytorch framework in Python to build the model. The specific training environment is configured as follows: CPU Inter(R) Core(TM) i7-6700HQ CPU@2.60GHz, Python version 3.8, Pytorch 1.10.0, and Windows 10.

This paper employs a simulation platform of the BFIP to authenticate the proposed method. The primary interface of the platform is depicted in Figure 7. This platform has the capability to simulate the values of process variables during fault occurrences and also includes interactive functionalities. It can simulate the changes in the BFIP in response to input control commands when faults occur.

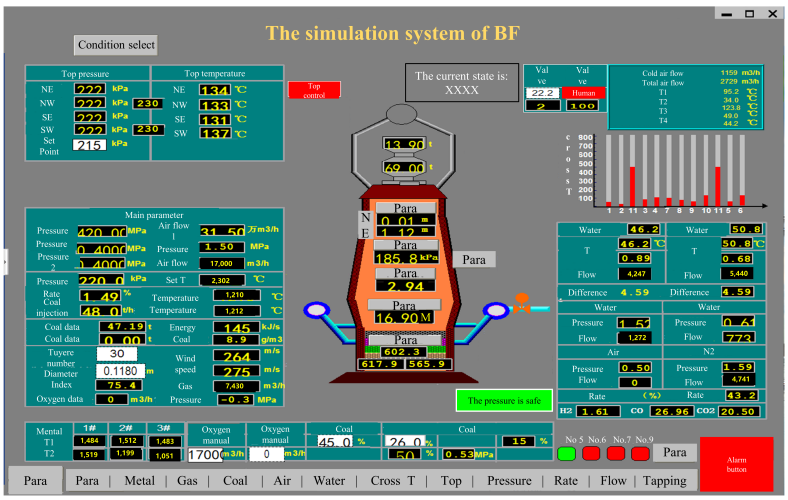

Figure 7. The simulation platform of the BFIP.

4.2. IKG Compeletion

To confirm the effectiveness of the graph embedding approach introduced in this paper, several traditional graph embedding methods were chosen for comparative analysis. A comprehensive listing of these methods and their respective characteristics are presented in Table 1.

In this study, we utilized a simulation platform to gather dynamic data from an IKG constructed for the BFIP. Through the creation of various failure scenarios within the BFIP system, we were able to capture the attributes and interconnections of nodes from the onset to the resolution of failures. A total of 51 nodes were implicated in this analysis. Specifically, we documented the state of nodes at 3500 time points across three distinct blast furnace failure events: low stockline, blowing out, and tuyere failure. The recorded data encompassed the attribute values of individual nodes and their connectivity status at each time point. We then used this recorded data for IKG compeletion experiments.

Table 1. All graph embedding methods in experiments and their characteristics.

Method	Dynamic	Semantic	Structure	Feature	Aggregation	Text
GraphSAGE	✗	✗	✓	✓	✓	✗
CTDNE	✓	✗	✓	✗	✗	✗
DySAT	✓	✗	✓	✓	✗	✗
TGAT	✓	✗	✓	✓	✓	✗
Two-stage model without text	✓	✓	✓	✓	✓	✗
Two-stage model	✓	✓	✓	✓	✓	✓

First, we undertook a transductive learning task to forecast future connections between nodes that had been identified in the training data using the aforementioned methods, which can be seen as the same as the IKG compeletion task. Four metrics, namely accuracy, average precision, recall rate, and F1 score, were chosen to assess their effectiveness. The outcomes are presented in Table 2.

Table 2. IKG compeletion task performance of graph embedding methods.

Method	Accury	Average Precision	Recall	F1
GraphSAGE	0.5175	0.5090	1	0.6745
CTDNE	0.7375	0.7262	0.6557	0.7375
DySAT	0.7575	0.6734	1	0.8048
TGAT	0.81	0.7246	1	0.8403
Two-stage model without text	0.85	0.7692	1	0.8696
Two-stage model	0.9125	0.8511	1	0.9195

As illustrated in Table 2, the static graph approach utilized by GraphSAGE demonstrates suboptimal performance in the prediction of edges within dynamic graphs. Conversely, dynamic graph embedding techniques generally display superior efficacy. Specifically, DySAT and CTDNE exhibit a higher recall rate compared to TGAT, suggesting a greater propensity to discern the absence of edges between nodes, potentially leading to the exclusion of significant node associations. In contrast, the method proposed in this study integrates text information and amalgamates both graph structure and node data to achieve more precise node relationship prediction, resulting in superior performance across all metrics compared to alternative methods.

Figure 8 depicts the spatial configuration of nodes derived from different graph embedding methods. In the static method space of GraphSAGE, the interconnections between nodes remain constant. Conversely, in the space produced by dynamic methods, the relative positioning of nodes fluctuates over time, thereby facilitating the dynamic forecasting of node relationships. Methods with inferior prediction outcomes exhibit more pronounced clustering among graph nodes, such as GraphSAGE, while dynamic embedding methods with superior prediction results disperse nodes more evenly and densely in the space, as demonstrated by TGAT and the two-stage model without text. The node distribution of the two-stage model resembles that of the two-stage model without text, but it is altered, indicating the evident influence of text information on it.

Additionally, we utilized the platform to assess the efficacy of the two-stage model in the context of self-healing control tasks within the BFIP, with a specific focus on low stockline scenarios.

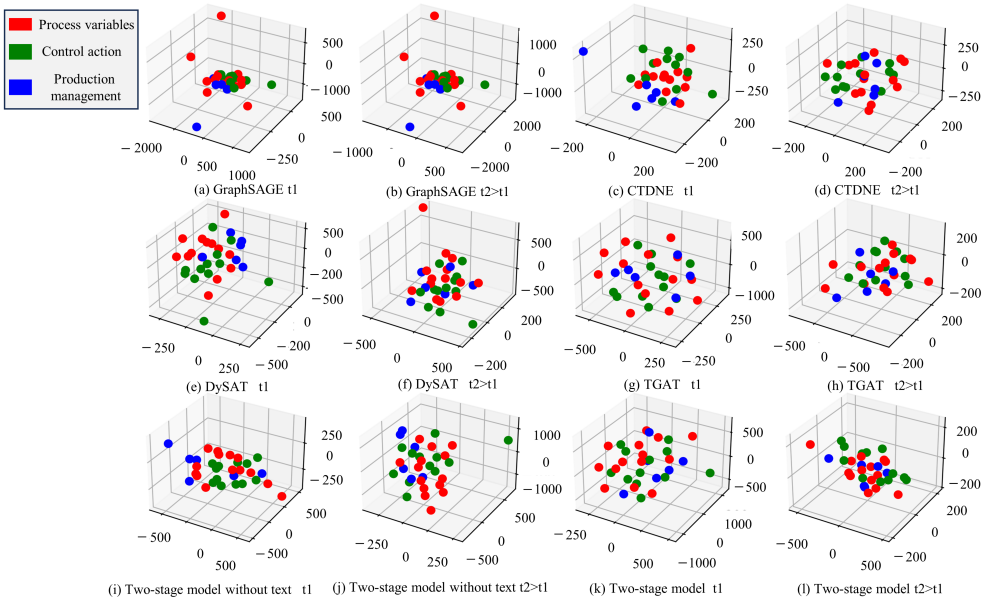

Figure 8. The spatial configuration of nodes derived from different graph embedding methods.

4.3. Self-Healing Control Task: Low Stockline

We used dynamic graph embedding methods in our simulation experiments to compare our proposed method, and proposed four quantitative metrics to measure the performance of these methods in the task of self-healing control in the BFIP, as follows:

- Whether the fault is eliminated: The value is 0 or 1, where 0 represents failure cannot be eliminated and 1 represents that the failure can be eliminated. This indicator is the most important indicator to measure whether the method can achieve self-healing control.
- Correct actions number: The number of correct control actions output during the control process.
- Incorrect actions rate: The proportion of the number of erroneous actions output during the control process to the total output control actions.
- Time to eliminate the fault: The total time required to eliminate the fault.

Based on the four metrics, we conducted statistical analysis on the performance of dynamic graph embedding methods and the method proposed in this paper in simulation experiments. Ten experiments were conducted for each method, with the best-performing outcome chosen for data collection. The experimental results are shown in Table 3.

Table 3. Self-healing control task performance of methods.

Method	Whether the Fault Is Eliminated	Correct Actions Number	Incorrect Actions Rate	Time to Eliminate the Fault (Min)
CTDNE	0	16 (28)	0.2728	Null
DySAT	0	20 (28)	0.2593	Null
TGAT	0	25 (28)	0.1935	Null
Two-stage model without text	0	28 (28)	0.1250	Null
Two-stage model	1	28 (28)	0	278

According to Table 3, we can see that the dynamic graph embedding methods cannot complete the self-healing control task in the BFIP. On the one hand, it is unable to output all the control actions required during the entire process. On the other hand, the occurrence of incorrect actions also leads to deviations in the overall control process, which is unacceptable for a high-safety industrial process such as the BFIP. Compared with two-stage model without text, the framework proposed in this paper can complete the self-healing control

of the BFIP because it can output all correct actions without outputting incorrect actions at the same time. This is due to the existence of text data, which enhances the exclusion of non-existent edges. Therefore, it can be seen that obtaining richer information from multi-source heterogeneous data can help improve the performance of the model.

The operational issue known as low stockline in the BFIP can be rectified by making adjustments to the smelting conditions. This particular fault necessitates a comprehensive series of control measures and has the longest control process, as illustrated in Figure 9. In the event of low stockline occurrence, the initial step is to close the lower sealing valve. Subsequently, the hot air pressure, top pressure, rich oxygen flow rate, and coal injection rate are gradually reduced to diminish the smelting intensity. Following this, the control commands involve opening the lower sealing valve and adding coke, while also adjusting the fabric angle. It is crucial to maintain this reduced intensity smelting state for a specific duration, during which the recovery of the stockline serves as an indicator to determine the resolution of the fault. Once the stockline has reached approximately 2.5 m, the smelting conditions can be restored to their original level. Figure 9c demonstrates that the top temperature increases when the lower sealing valve is closed and returns to normal when the valve is opened. The permeability remains stable, while the flow rate of cold air and total pressure difference fluctuate with changes in hot air pressure and top pressure. Therefore, the proposed method achieves a comprehensive self-healing control process.

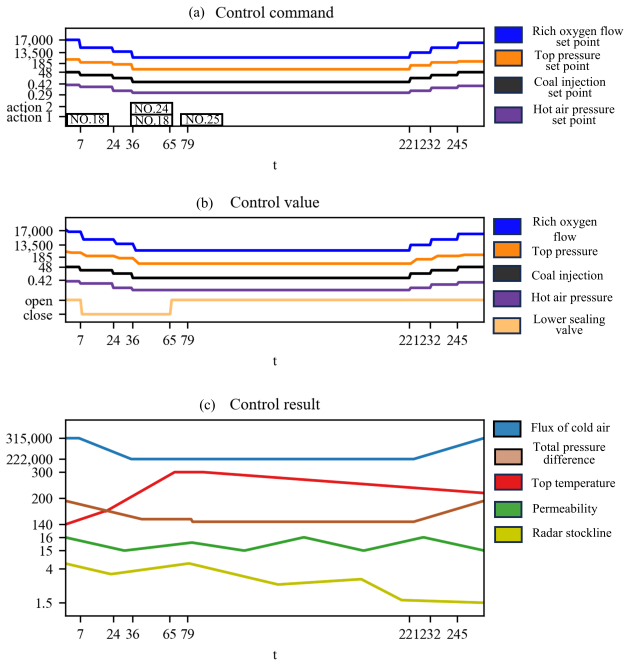

Figure 9. The self-healing control process of low stockline based on the two-stage model. (The coordinates of the y-axis of the graph have been deformed, the values of the different curves on the same graph are based on the values of the y-coordinate labeled value).

5. Conclusions

In the present era, the challenges of updating entity triples in a graph using continuous production data to maximize knowledge coverage and applying knowledge graphs to address diverse industrial tasks are significant issues for IKGs. This paper introduces a two-stage model construction approach aimed at enhancing both knowledge graph completion and industrial tasks. Initially, this paper outlines the various forms of multi-source data in the industry and presents processing techniques for each data type, emphasizing data vectorization and conceptual alignment to achieve multi-source data fusion modeling. Subsequently, the paper delineates two interconnected subtasks for developing a pretrained language–knowledge graph model through multi-task learning. Additionally, to account for the dynamic nature of the production process, a dynamic expert network structure is

integrated with the pretrained model for different tasks. The experimental results indicate that the proposed approach outperforms some classic methods in terms of both accuracy and practical applicability.

However, the shortcomings of this paper are listed as follows:

- The limitation of aligning text with an IKG currently involves using named entity recognition from text and using a word bank constructed from the IKG, which may lead to insufficient mining of text data.
- There is a lack of certain strategies for the dynamic updating of IKGs, and when multiple tasks update the IKG simultaneously, there may be conflicts, resulting in negative information transfer.

Therefore, future research directions can involve the following:

- Extracting triples from text to automatically enable better interaction between text and the IKG.
- Research on continuous learning of an IKG and developing knowledge graph updating strategies under multi-task conditions.

In the future, we will collect more production data for training the model, with the goal of achieving a level comparable to that of larger models. This will enable the development of large-scale models within specific vertical domains.

Author Contributions: All authors contributed to the writing and revisions; conceptualization, X.H. and C.Y.; methodology, X.H.; software, X.H.; validation, X.H. and C.Y.; data curation, X.H.; visualization, X.H.; supervision, C.Y.; funding acquisition, C.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China grant number 61933015, 61903326.

Data Availability Statement: Data are not available due to restrictions of privacy.

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

References

1. Li, X.; Lyu, M.; Wang, Z.; Chen, C.H.; Zheng, P. Exploiting knowledge graphs in industrial products and services: A survey of key aspects, challenges, and future perspectives. *Comput. Ind.* **2021**, *129*, 103449. [CrossRef]
2. Cao, Q.; Giustozzi, F.; Zanni-Merk, C.; de Bertrand de Beuvron, F.; Reich, C. Smart condition monitoring for industry 4.0 manufacturing processes: An ontology-based approach. *Cybern. Syst.* **2019**, *50*, 82–96. [CrossRef]
3. Poveda-Villalón, M.; Fernández-Izquierdo, A.; Fernández-López, M.; García-Castro, R. LOT: An industrial oriented ontology engineering framework. *Eng. Appl. Artif. Intell.* **2022**, *111*, 104755. [CrossRef]
4. Lou, S.; Yang, C.; Wu, P.; Yang, Y.; Kong, L.; Zhang, X. Data-Driven Joint Fault Diagnosis Based on RMK-ASSA and DBSKNet for Blast Furnace Iron-Making Process. *IEEE Trans. Autom. Sci. Eng.* **2023**, *2023*, 1–16. [CrossRef]
5. Kong, L.; Yang, C.; Lou, S.; Cai, Y.; Huang, X.; Sun, M. Collaborative Extraction of Intervariable Coupling Relationships and Dynamics for Prediction of Silicon Content in Blast Furnaces. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 1–13. [CrossRef]
6. Huang, X.; Yang, C.; Zhang, H.; Lou, S.; Dali, G.; Kong, L. Data and knowledge collaborative-driven fault identification and self-healing control action inference framework for blast furnace. *IEEE Trans. Autom. Sci. Eng.* **2024**, *245*, 123040. [CrossRef]
7. Lyu, M.; Li, X.; Chen, C.H. Achieving Knowledge-as-a-Service in IIoT-driven smart manufacturing: A crowdsourcing-based continuous enrichment method for Industrial Knowledge Graph. *Adv. Eng. Inform.* **2022**, *51*, 101494. [CrossRef]
8. Xia, L.; Liang, Y.; Leng, J.; Pai, Z. Maintenance planning recommendation of complex industrial equipment based on knowledge graph and graph neural network. *Reliab. Eng. Syst. Saf.* **2023**, *232*, 109068. [CrossRef]
9. Shen, T.; Zhang, F.; Cheng, J. *A Comprehensive Overview of Knowledge Graph Completion*; Elsevier: Amsterdam, The Netherlands, 2022; Volume 255, p. 109597.
10. Wang, Z.; Zhang, B.; Gao, D. A novel knowledge graph development for industry design: A case study on indirect coal liquefaction process. *Comput. Ind.* **2022**, *139*, 103647. [CrossRef]
11. Li, X.; Zheng, P.; Bao, J.; Gao, L.; Xu, X. Achieving cognitive mass personalization via the self-X cognitive manufacturing network: An industrial knowledge graph-and graph embedding-enabled pathway. *Engineering* **2023**, *22*, 14–19. [CrossRef]
12. Kim, B.; Hong, T.; Ko, Y.; Seo, J. Multi-task learning for knowledge graph completion with pre-trained language models. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain (Online), 8–13 December 2020; pp. 1737–1743.

13. Yang, L.; Xiao, Z.; Jiang, W.; Wei, Y.; Hu, Y.; Wang, H. Dynamic heterogeneous graph embedding using hierarchical attentions. In Proceedings of the Advances in Information Retrieval: 42nd European Conference on IR Research, Lisbon, Portugal, 14–17 April 2020; pp. 425–432.
14. Sankar, A.; Wu, Y.; Gou, L.; Zhang, W.; Yang, H. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In Proceedings of the 13th International Conference on Web Search And Data Mining, Houston, TX, USA, 3–7 February 2020; pp. 519–527.
15. Huang, H.; Shi, R.; Zhou, W.; Wang, X.; Jin, H.; Fu, X. Temporal Heterogeneous Information Network Embedding. In Proceedings of the International Joint Conference on Artificial Intelligence, Virtual, 19–26 August 2021; pp. 1470–1476.
16. Zhou, W.; Huang, H.; Shi, R.; Song, X.; Lin, X.; Wang, X.; Jin, H. Temporal Heterogeneous Information Network Embedding via Semantic Evolution. *IEEE Trans. Knowl. Data Eng.* **2023**, *2023*, 1–12. [CrossRef]
17. Sankar, A.; Wu, Y.; Gou, L.; Zhang, W.; Yang, H. Dynamic graph representation learning via self-attention networks. *arXiv* **2018**, arXiv:1812.09430.
18. Gao, M.; Li, J.Y.; Chen, C.H.; Li, Y.; Zhang, J.; Zhan, Z.H. Enhanced multi-task learning and knowledge graph-based recommender system. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 10281–10294. [CrossRef]
19. Wang, H.; Zhang, F.; Zhao, M.; Li, W.; Xie, X.; Guo, M. Multi-task feature learning for knowledge graph enhanced recommendation. In Proceedings of the World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 2000–2010.
20. Cao, Q.; Beden, S.; Beckmann, A. A core reference ontology for steelmaking process knowledge modelling and information management. *Comput. Ind.* **2022**, *135*, 103574. [CrossRef]
21. Dobrev, M.; Gocheva, D.; Batchkova, I. An ontological approach for planning and scheduling in primary steel production. In Proceedings of the International IEEE Conference Intelligent Systems, Beijing, China, 12–15 October 2008; pp. 6–14.
22. Muñoz, E.; Capón-García, E.; Moreno-Benito, M.; Espuña, A.; Puigjaner, L. Scheduling and control decision-making under an integrated information environment. *Comput. Chem. Eng.* **2011**, *35*, 774–786. [CrossRef]
23. Wang, X.; Wong, T.; Fan, Z.P. Ontology-based supply chain decision support for steel manufacturers in China. *Expert Syst. Appl.* **2013**, *40*, 7519–7533. [CrossRef]
24. Yasunaga, M.; Bosselut, A.; Ren, H.; Zhang, X.; Manning, C.; Liang, P.; Leskovec, J. Deep bidirectional language-knowledge graph pretraining. *Adv. Neural Inf. Process.* **2022**, *35*, 37309–37323.
25. Zhang, X.; Bosselut, A.; Yasunaga, M.; Ren, H.; Liang, P.; Manning, C.D.; Leskovec, J. GreaseLM: Graph REASoning Enhanced Language Models. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022.