Intelligent Human Systems Integration (IHSI), Vol. 200, 2026, 274–284
https://doi.org/10.54941/ahfe1007087

Knowledge Engineering With Large Language Models: Accelerating Fuzzy Rule Bases Development for Energy-Aware Expert Systems

Borys Ioshchikhes, Ann-Kathrin Bischoff, Jerome Stock, Michael Frank, and Matthias Weigold

Technical University of Darmstadt, Institute for Production Management, Technology and Machine Tools (PTW), 64287 Darmstadt, Germany

ABSTRACT

Expert systems offer a promising way to automatically identify energy efficiency potentials in industry and thereby contribute to energy cost savings and decarbonization. In these systems, domain-specific knowledge is embedded and linked to automated analyses of measurement data. Until now, knowledge engineers have extracted, structured, and represented the necessary domain-specific knowledge in a form usable by expert systems, which is time-consuming and costly. This article presents a hybrid approach that couples expert systems with large language models to support the work of knowledge engineers. Energy performance indicators, selected by the energy manager, serve to quantify changes in energy performance and reproduce the heuristic decision-making of human experts on a quantitative basis. These indicators then form the basis for a rule set that targets areas with the highest potential energy savings. For practical implementation, a fuzzy rule base is applied because it captures decisions made under imprecise information and allows conditions and conclusions that can be partially true or false. Building the fuzzy rule base involves assigning membership functions to input and output variables and defining their linguistic partitioning, since these choices shape both sensitivity and interpretability. The rule base is implemented as generally understandable IF–THEN rules. The premise consists of energy performance indicators that are associated with linguistic variables and combined using logical operators. The conclusion contains priority numbers, which are also associated with linguistic variables and express the energy efficiency potential. In the hybrid setup presented in this article, large language models formalize given energy performance indicators and fuzzy rules, propose membership functions to populate the fuzzy rule base, and generate visualization scripts in Python. This leads to accelerated development while preserving transparent, comprehensible, and reproducible decision logic characteristic of expert systems. The approach is demonstrated using a foam panel production line in the chemical industry.

Keywords: Energy analysis, Knowledge-based systems, Artificial intelligence, Climate neutrality

INTRODUCTION

The industrial sector plays a decisive role in achieving climate targets, as it is the largest emitter of energy-related greenhouse gases in the European Union (Hannah Ritchie, 2020). At the same time, industrial companies must

Received September 9, 2025; Revised November 10, 2025; Accepted November 25, 2025; Available online February 1, 2026

ensure their global competitiveness. In this tension field, energy efficiency is regarded as a key lever, since it contributes directly to emission reduction while allowing production costs to be reduced (European Parliament, Council of the European Union, 2023). Despite its relevance, the identification of energy efficiency potentials in manufacturing is often constrained by organizational and knowledge-related barriers, hindering subsequent realization efforts. Surveys reveal that companies frequently lack awareness of suitable measures, digital competencies among staff remain limited, and consulting resources are scarce (ABB Ltd, 2022).

Expert systems (ESs) offer a promising way to address these obstacles and support industrial companies in achieving both cost savings and decarbonization goals. ESs are computer programs designed to replicate the reasoning of human experts by integrating domain-specific knowledge with automated analysis of measurement data. In the context of energy management, such systems systematically identify efficiency potentials and guide decision-making with fuzzy logic for improvement measures (Seyfried et al., 2024). The development of ESs, however, relies on knowledge engineers, who extract, structure, and formalize expert knowledge into machine-usable representations. This process is labor-intensive and often limits the scalability of ESs across industrial domains (Ioshchikhes & Zink et al., 2025). To reduce the dependency on manual knowledge engineering and to accelerate ES development, this paper proposes a hybrid approach that employs large language models (LLMs) as assistants in the formalization process. A complete substitution of conventional ESs is not intended. While LLMs can also enhance energy efficiency, ESs remain essential for ensuring transparency, reproducibility, and deterministic reasoning based on explicitly formalized knowledge (Wu et al., 2025). In contrast, LLMs operate probabilistically and generate responses without guaranteed consistency or intrinsic explainability (Martens & Cap, 2025). Combining both paradigms allows the transparency of ESs to be maintained while leveraging the generative strengths of LLMs.

Recent works addressing knowledge engineering with LLMs have increasingly explored how such models can support the extraction and structuring of expert knowledge for rule-based or symbolic systems. Duranti et al. (2025) combine LLM outputs with symbolic verification modules to derive linear temporal and description logic specifications from semi-structured textual sources, ensuring consistency through formal entailment checking. Similarly, Chen et al. (2025) integrate LLM-based semantic guidance with diffusion modeling to generate and refine rule sets over temporal knowledge graphs, demonstrating how probabilistic language models can contribute to interpretable rule generation. Within the field of fuzzy modeling, E et al. (2022) investigate the construction of fuzzy rule-based systems through fuzzy relational factorization to reduce dimensionality and improve interpretability. Yet these approaches remain primarily data-driven and do not leverage natural-language input. Although these contributions advance automated rule induction and hybrid neuro-symbolic reasoning, they predominantly address formal logic, graph reasoning, or data-centric contexts. They do not focus on the domain-specific knowledge engineering processes required for ESs.

In particular, existing research does not focus on the translation of natural-language heuristics into fuzzy rules or on the semi-automatic design of membership functions and linguistic partitions that maintain expert interpretability. Against this background, this paper presents a hybrid approach integrating LLMs into the process of developing fuzzy rule bases for energy-aware ESs used to improve energy efficiency in manufacturing.

KNOWLEDGE ENGINEERING WORKFLOW

Following the introduction, this section describes the hybrid workflow for LLM-assisted knowledge engineering in the development of fuzzy rule bases for energy-aware ESs.

Workflow Overview

Figure 1 shows the overall workflow. LLM-supported functions are highlighted in green, and tasks performed by the energy manager, who coordinates energy management activities, are highlighted in gray. Following the identification of energy relevant machines, energy performance indicators (EnPIs) are set in accordance with ISO 50006 (A1), which form the quantitative basis for evaluating energy efficiency potentials (International Organization for Standardization, 2023). These indicators are selected by the energy manager in collaboration with process experts. In the next step, the LLM formalizes natural-language descriptions of EnPIs into structured, machine-interpretable representations (A2). The formalized EnPIs are then utilized to formulate a fuzzy rule base (A3). The fuzzy approach assumes that human experts often make decisions without precisely quantified information and therefore accounts conditions or conclusions that may be partially true or false (Liao, 2005). The rules are structured as IF-THEN statements. This ensures a machine-readable and applicable representation and allows for explicit and comprehensible functional transparency. The premise consists of EnPIs, which are assigned linguistic variables and linked using logical operators. The conclusion contains priority numbers, which are also assigned linguistic variables and represent a measure of the unrealized energy efficiency potential. The formulated rules are then formalized by the LLM (A4), which translates the verbal logic into consistent fuzzy syntax, verifies linguistic mappings, and validates logical coherence. Furthermore, the LLM proposes suitable membership functions for the fuzzy variables, such as triangular or trapezoidal shapes, and suggests initial parameter values (A5). Finally, the workflow concludes with an automatic visualization stage (A6), in which the LLM generates a Python script to plot membership functions and fuzzy inference surfaces. These visualizations support validation and communication between the energy manager and knowledge engineer, ensuring that the resulting rule base behaves transparently and remains explainable.

While the LLM accelerates repetitive formalization tasks and provides structured proposals, a knowledge engineer remains in the loop to critically review, validate, and refine these outputs, ensuring semantic precision and domain reliability.

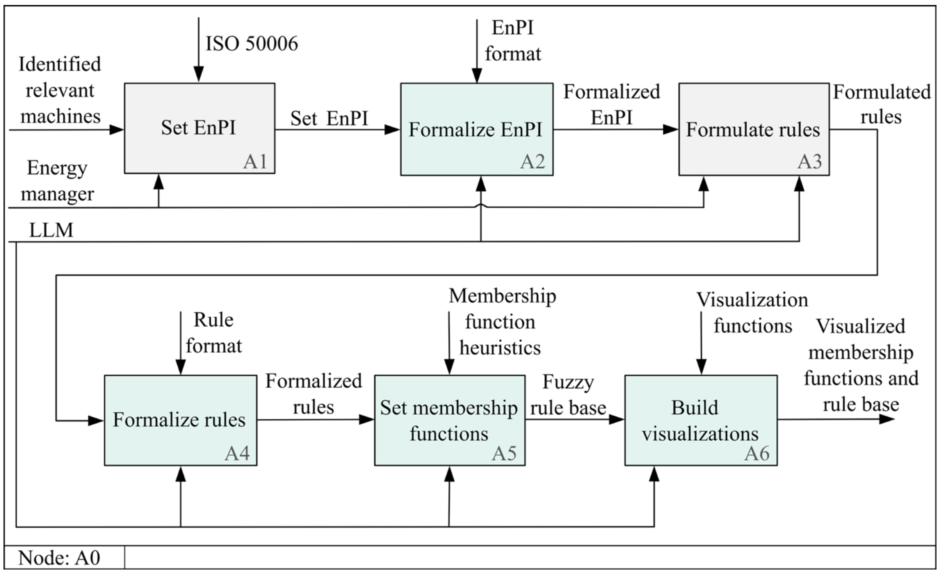

Figure 1: SADT diagram of the LLM-assisted knowledge-formalization workflow.

Code-Generation Pipeline

To operationalize the workflow described above, the code-generation pipeline (see Figure 2) transforms the specifications into importable Python modules. Specifications are provided either via a command-line wizard or by editing two canonical Markdown files (enpis.md, rules.md), which together define EnPI names, symbols, and formulas as well as fuzzy variables, linguistic partitions, and IF–THEN rules. From this input, the generator builds a schema-constrained prompt that fixes function signatures, imports, file layout, and naming conventions while instructing the model to emit code only. Inference produces an in-memory Python draft for the target artefact (e.g., EnPIs.py, FIS.py, or visualizer.py) under conservative decoding settings that prioritise complete, reproducible outputs.

Each draft then passes a syntactic and interface gate. The pipeline compiles the returned text in memory using Python's built-in `compile()` function, which parses the source and returns a compiled representation suitable for execution. If a syntax error is detected, the draft is discarded and a deterministic implementation is rendered from the same Markdown specifications. This ensures that valid modules are produced even when the LLM emits an incomplete snippet and keeps the process robust for non-specialists. In both the LLM and fallback paths, lightweight interface checks verify the required callables. Provenance is preserved by archiving the raw model draft and a short generation log, while only the accepted, syntactically valid module is written for downstream use. The resulting files enable direct integration into the *Expert System Shell for Energy Efficiency* (*ESS4EE*) to build an energy-aware ES (Ioshchikhes & Frank et al., 2025).

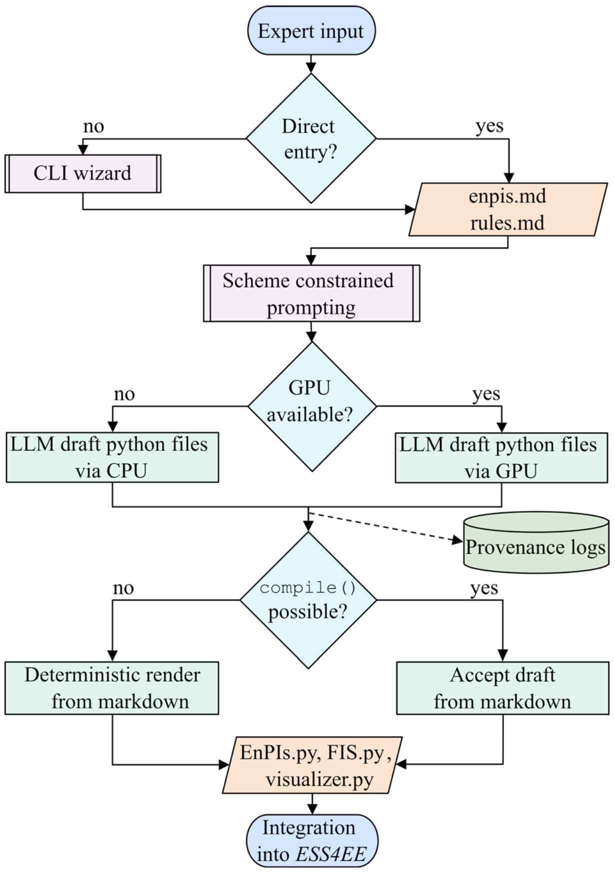
Expert input
Direct entry?
no
yes
CLI wizard
enpis.md
rules.md
Scheme constrained prompting
GPU available?
no
yes
LLM draft python files via CPU
LLM draft python files via GPU
Provenance logs
compile() possible?
no
yes
Deterministic render from markdown
Accept draft from markdown
EnPIs.py, FIS.py, visualizer.py
Integration into ESS4EE

Figure 2: Flowchart of the on-prem code-generation pipeline.

Large Language Model Setup and Configuration

The language model in this study runs entirely on premises to meet confidentiality, reproducibility, and predictable latency requirements. Its role is schema-constrained code synthesis. From concise, structured specifications of EnPIs and fuzzy rules, it produces self-contained Python modules that compile without manual editing. We selected the model *Microsoft Phi-3-mini-4k-instruct* because it balances capability and efficiency for constrained scenarios (Abdin et al., 2024). The model has about three to four billion parameters and a context window of roughly 4,000 tokens, which is sufficient for our prompts and the small amount of in-prompt guidance used to enforce function signatures and file structure.

The inference is executed locally. On Windows hosts with AMD graphics hardware, the model runs through ONNX Runtime GenAI with DirectML. ONNX Runtime GenAI is a high-performance execution engine for generative neural networks. It loads a serialized model graph in the open ONNX format and runs it with optimized kernels. On Windows, DirectML connects this engine to the installed GPU driver so the same model runs accelerated on a wide range of consumer hardware without vendor-specific code (Microsoft, 2025). If a suitable GPU is not present, the same prompts and decoding settings are executed through a Hugging Face Transformers stack on the CPU. Hugging Face Transformers is a widely used open-source library that provides model definitions, tokenizers, and generation routines for many architectures (Jain, 2022). In our setup it supplies the tokenizer and a reliable CPU inference path, so behavior and outputs remain consistent across machines. After a one-time download of the model snapshot, the pipeline operates offline and no inputs, prompts, or outputs leave the host system.

The generation process is configured for correctness and stability. We use a low sampling temperature and a bounded maximum number of new tokens so that the model favors complete outputs over creative variation. The prompts are scheme-constrained. They specify function signatures, required imports, and file layout, and explicitly instruct the model to return code only. To reduce memory use and improve inference efficiency, measured as higher generated-tokens-per-second, the model's weights are quantized to four bits using activation-aware weight quantization (AWQ). This method takes activation statistics into account so that accuracy is preserved even when weights are stored with four-bit precision. In practice this keeps instruction-following quality high while lowering the memory footprint to a few gigabytes. Table 1 summarizes the LLM specifications.

Table 1: Technical summary of the local LLM configuration.

Model	**Microsoft Phi-3-mini-4k-instruct**
Parameter scale	~3-4B parameters
Context window	~4,000 tokens
Quantization	INT4 (AWQ); weight-only
Runtime backends	ONNX Runtime GenAI (DirectML); CPU fallback via Hugging Face Transformers
Deployment	Fully on-prem; offline after initial snapshot
Decoding policy	Low temperature; bounded max-new-tokens
Prompt design	Schema-constrained templates
Reliability check	Python `compile()` on each draft; deterministic template fallback on error
Privacy	No task data leaves the host machine
Memory	~2–3 GB for INT4 weights (plus runtime overhead)

CASE STUDY: FOAM PANEL PRODUCTION LINE

The workflow shown in Figure 1 is demonstrated using the example of a foam panel production line. During this process, polymers are expanded by heat to produce foam cores for applications in aviation. The polymer panels pass through four consecutive zones: First, they are led into a preheating cabinet. In the preheating zone and in the two subsequent zones, resistance heaters heat the chamber air, while fans ensure its circulation. The actual foaming takes place in the subsequent foam cabinet, where the panels expand to roughly ten times their original volume. This expansion creates internal stresses that cause deformations. In the next straightening cabinet, these stresses are reduced, and deformations minimized by selectively activating infrared heaters. Finally, the panels are flattened in a hydraulic press. The overall material flow of the process is shown in Figure 3.

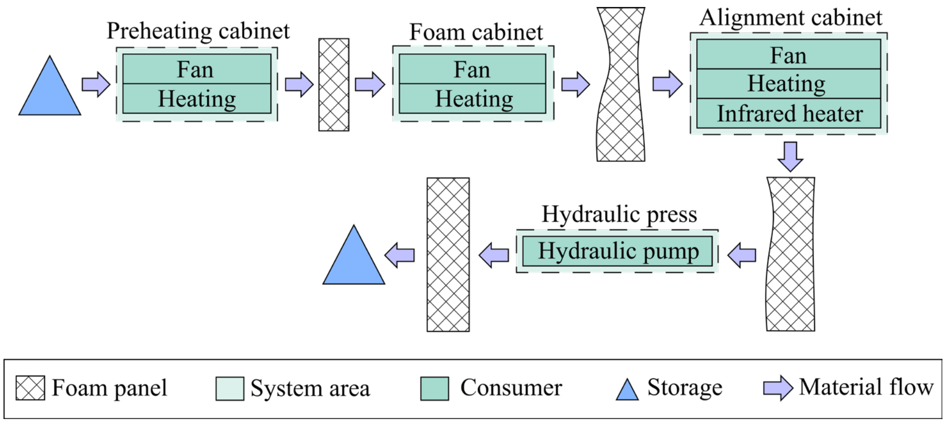

Figure 3: Material flow of the foam panel production line. Illustration based on (Erlach & Westkämper, 2009).

The starting point for identifying energy-efficiency potentials is the measured active electrical power of the individual consumers. In this use case, the energy-relevant information is the unproductive state, i.e., periods in which the consumers do not contribute to value creation. Detection of the unproductive state is performed by an algorithm that is not detailed here but is available in the full ES implementation (Ioshchikhes, 2025b).

To quantify the unproductive state, two EnPIs are defined: the non-productive time factor (*NPTF*) and the non-productive energy factor (*NPEF*) (Dehning et al., 2019). The *NPTF* is the ratio of non-productive time t_{np} to the total observation time t_{total},

$$NPTF = \frac{t_{\text{np}}}{t_{\text{total}}} \cdot 100\ \%,$$

and the NPEF is the ratio of non-productive energy E_{np} to total energy E_{total}:

$$NPEF = \frac{E_{\text{np}}}{E_{\text{total}}} \cdot 100\ \%.$$

For both EnPIs, values near 0 % indicate a high share of value-adding operation, whereas values near 100 % indicate substantial potential for improvement.

After the EnPIs are specified in Markdown or via the command-line wizard (see Figure 2), the fuzzy rule base is defined. Using *NPTF* and *NPEF* as inputs and the rule base given in Table 2, the priority score ζ is computed by Mamdani inference (Mamdani & Assilian, 1975). The priority score is the result of the fuzzy system, which represents the extent of the unrealized energy efficiency potential. It takes values in the interval [0, 1]. Values close to 1 indicate a higher energy-efficiency potential.

Table 2: Rule base for the ES of the foam panel production line.

Premise (IF)	Consequent (THEN)
NPTF is high AND *NPEF* is high	ζ is high
NPTF is medium AND *NPEF* is high	ζ is high
NPTF is low AND *NPEF* is high	ζ is high
NPTF is high AND *NPEF* is medium	ζ is high
NPTF is medium AND *NPEF* is medium	ζ is high
NPTF is low AND *NPEF* is medium	ζ is medium
NPTF is high AND *NPEF* is low	ζ is low
NPTF is medium AND *NPEF* is low	ζ is low
NPTF is low AND *NPEF* is low	ζ is low

Finally, the rule base together with the membership functions are formalized into Python code by the LLM. The generated visualization routine produces Figure 4. It visualizes the membership functions of the input and output variables and the resulting priority surface from the rule base, enabling experts to check that the linguistic partitions and IF–THEN rules yield the intended behavior.

EnPIs are computed for the observation period as specified in Table 3. The results assign high priority for efficiency improvement to the fan of the foam cabinet and to the heating of the straightening cabinet. They also indicate that opportunities for improvement arise in the heating of the preheating cabinet and the foam cabinet. Lower relevance for energy optimization measures is indicated for the fans in the preheating and straightening cabinet, as well as the infrared heating.

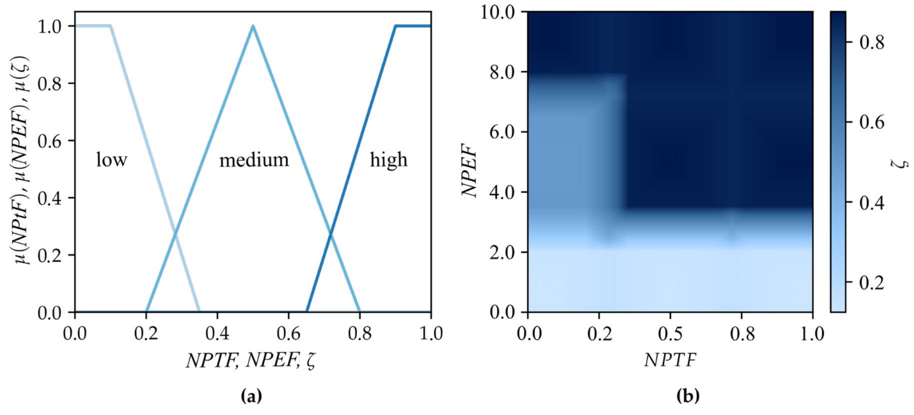

Figure 4: (a) Visualized membership functions and (b) rule base.

Table 3: Calculated EnPIs for the foam panel production line.

Consumer	t_{total} in h	t_{np} in h	E_{total} in kWh	E_{np} in kWh	*NPTF* in %	*NPTF* in %	ζ in 1
Preheating fan				192.3		2.1	0.4
Straightening fan				88.9		0.9	0.2
Foam fan				755.6		8.3	0.9
Straightening heating	168.0	41.0	9116.4	516.9	0.2	5.7	0.6
Infrared heating				12.3		0.1	0.2
Foam heating				280.8		3.1	0.6
Preheating heating				297.1		3.3	0.6

CONCLUSION

This paper introduced a hybrid workflow that integrates LLMs into the development of fuzzy rule bases for energy-aware ESs. The approach accelerates knowledge formalization by enabling automated generation of membership functions, rule syntax, and visualization scripts, while preserving the transparency and interpretability of rule-based reasoning. Implemented fully on premises, the approach ensures confidentiality and consistent behavior across different hardware setups. The case study of a foam panel production line demonstrated that the LLM-assisted pipeline can reliably produce executable modules for energy performance evaluation and effectively highlight areas with high energy efficiency potential.

The LLM employed in this study served as a proof of concept and was not optimized or validated across diverse industrial scenarios. Future work will therefore focus on benchmarking and fine-tuning different models to improve reliability and domain transferability. Additional research will investigate adaptive feedback mechanisms involving human experts and operational data to iteratively refine the generated rule bases. Further integration with the *ESS4EE* will enable scalable deployment across production environments and support continuous improvement toward data-driven, transparent, and energy-efficient manufacturing.

CREDIT AUTHOR STATEMENT

Borys Ioshchikhes: Conceptualization, Methodology, Investigation, Software, Validation, Formal Analysis, Data Curation, Writing - Original Draft, Visualization. **Ann-Kathrin Bischoff:** Conceptualization, Writing - Original Draft. **Jerome Stock**: Writing - Review and Editing. **Michael Frank:** Data Curation. **Matthias Weigold:** Supervision, Project administration, Fund acquisition.

DATA AVAILABILITY STATEMENT

The code for the code-generation pipeline presented in Figure 2 is available at Ioshchikhes (2025a). The *ESS4EE*, which supports building energy-aware

ESs across different use cases, is available at Ioshchikhes and Yoldas et al. (2025). A complete implementation of the ES for the demonstrated foam panel production line, including the measurement data, is available at Ioshchikhes (2025b).

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support from the German Federal Ministry of Economic Affairs and Energy (BMWE) for the project *etaGPT* (grant agreement No. 03EN4107B) and project supervision by the project management Jülich (PtJ).

REFERENCES

ABB Ltd (2022) Accelerating Ambition: How global industry is speeding up investment in energy efficiency, Helsinki. Available from: https://www.energyefficiencymovement.com/wp-content/uploads/2022/04/ABB-Energy-Efficiency-Survey-Report-2022.pdf [Accessed 9 June 2025].

Abdin, M., Aneja, J., Awadalla, H., Awadallah, A., Awan, A.A. & Bach, N. et al. (2024) Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. Available from: http://arxiv.org/pdf/2404.14219v4.

Chen, K., Song, X., Wang, Y., Gao, L., Li, A. & Zhao, X. et al. (2025) LLM-DR: A Novel LLM-Aided Diffusion Model for Rule Generation on Temporal Knowledge Graphs. Proceedings of the AAAI Conference on Artificial Intelligence, 39(11), 11481–11489. Available from: https://doi.org/10.1609/aaai.v39i11.33249.

Dehning, P., Blume, S., Dér, A., Flick, D., Herrmann, C. & Thiede, S. (2019) Load profile analysis for reducing energy demands of production systems in non-production times. Applied Energy, 237, 117–130. Available from: https://doi.org/10.1016/j.apenergy.2019.01.047.

Duranti, D., Giorgini, P., Mazzullo, A., Robol, M. & Roveri, M. (2025) LLM-Driven Knowledge Extraction in Temporal and Description Logics. In: Alam, M., Rospocher, M., van Erp, M., Hollink, L. & Gesese, G.A. (Eds.) Knowledge Engineering and Knowledge Management. Springer Nature Switzerland: Cham, pp. 190–208.

E. H., Cui, Y., Pedrycz, W., Robinson Fayek, A., Li, Z. & Li, J. (2022) Design of fuzzy rule-based models with fuzzy relational factorization. Expert Systems with Applications, 206, 117904. Available from: https://doi.org/10.1016/j.eswa.2022.117904.

Erlach, K. & Westkämper, E. (Eds.) (2009) Energiewertstrom: Der Weg zur energieeffizienten Fabrik. Fraunhofer Verl.: Stuttgart.

European Parliament, Council of the European Union (2023) Directive (EU) 2023/1791 of the European Parliament and of the Council of 13 September 2023 on energy efficiency and amending Regulation (EU) 2023/955 (recast) (Text with EEA relevance). Available from: http://data.europa.eu/eli/dir/2023/1791/oj [Accessed 17 October 2025].

Hannah Ritchie (2020) Sector by sector: where do global greenhouse gas emissions come from? Our World in Data. Available from: https://ourworldindata.org/ghg-emissions-by-sector [Accessed 17 October 2025].

International Organization for Standardization (2023) Energy management systems: Evaluating energy performance using energy performance indicators and energy baselines, Berlin. DIN Media GmbH.

Ioshchikhes, B. (2025a) Energy-Aware Expert System Code Generator, 19 October. Available from: https://git.ptw.maschinenbau.tu-darmstadt.de/eta-fabrik/publications/energy-aware-expert-system-code-generator [Accessed 17 October 2025].

Ioshchikhes, B. (2025b) Ergänzendes Material: Expertensysteme zur Identifikation und Bewertung von Energieeffizienzpotenzialen in der Fertigung. Available from: https://tudatalib.ulb.tu-darmstadt.de/handle/tudatalib/4224.4. Available from: https://doi.org/10.48328/tudatalib-1427.

Ioshchikhes, B., Frank, M., Joseph, T.M. & Weigold, M. (2025) Improving Energy Efficiency in Manufacturing: A Novel Expert System Shell. Procedia CIRP, 134, 615–620. Available from: https://doi.org/10.1016/j.procir.2025.02.218.

Ioshchikhes, B., Yoldas, C., Horn, L., Asl, N.A., Zimmer, L. & Wigandt, A. (2025) Expert System Shell for Energy Efficiency (ESS4EE). Available from: https://github.com/Borika95/ESS4EE.

Ioshchikhes, B., Zink, R., Ozen, O. & Weigold, M. (2025) A Holistic Framework for Developing Expert Systems to Improve Energy Efficiency in Manufacturing. Energies, 18(6), 1406. Available from: https://doi.org/10.3390/en18061406.

Jain, S.M. (2022) Hugging Face. In: Jain, S.M. (Ed.) Introduction to Transformers for NLP. Apress: Berkeley, CA, pp. 51–67.

Liao, S.-H. (2005) Expert system methodologies and applications—a decade review from 1995 to 2004. Expert Systems with Applications, 28(1), 93–103. Available from: https://doi.org/10.1016/j.eswa.2004.08.003.

Mamdani, E.H. & Assilian, S. (1975) An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies, 7(1), 1–13. Available from: https://doi.org/10.1016/S0020-7373(75)80002-2.

Martens, A. & Cap, C.H. (Eds.) (2025) Schreibende KI - ein interdisziplinärer Diskurs: Perspektiven über den Sinn oder Unsinn von schreibender KI. Springer Vieweg: Wiesbaden.

Microsoft (2025) ONNX Runtime GenAI: Generative AI extensions for onnxruntime. Available from: https://github.com/microsoft/onnxruntime-genai [Accessed 15 October 2025].

Seyfried, S., Kohne, T., Weyand, A., Ozen, O., Sossenheimer, J. & Zink, R. et al. (2024) 12 Energieeinsatz im Kontext einer klimaneutralen Produktion. In: Zäh, M.F. (Ed.) Handbuch Nachhaltige Produktion. Carl Hanser Verlag GmbH & Co. KG: München, pp. 307–344.

Wu, T., Li, J., Bao, J. & Liu, Q. (2025) Large language model-driven multi-agent systems for improving production efficiency and reducing carbon emissions in manufacturing. Computers & Industrial Engineering, 207, 111299. Available from: https://doi.org/10.1016/j.cie.2025.111299.