

Neuro-Symbolic AI for BPM

A Tutorial on Predictive and Prescriptive Process Analytics

Gyunam Park¹ and Fabrizio De Santis²

¹ Eindhoven University of Technology, Eindhoven, Netherlands
g.park@tue.nl

² University of Parma, Parma, Italy
fabrizio.desantis@unipr.it

Abstract. Process analytics has become predominantly data-driven: deep sequence models such as LSTMs and Transformers now dominate predictive and prescriptive process monitoring and deliver strong average-case accuracy. These models are, however, opaque, require large amounts of data, and do not incorporate the process knowledge that organizations already hold in the form of regulations, declarative constraints, and reference process models. The symbolic side of Business Process Management (BPM) carries exactly this knowledge but is brittle on noisy, real-world event data. *Neuro-symbolic AI* (NeSy) aims to bridge these two traditions, and a growing body of BPM work has begun to use it. This tutorial gives the BPM community an accessible overview of the relationship between BPM and NeSy. It introduces Kautz’s architectural taxonomy of NeSy and the *Learning from Logical Constraints* paradigm together with its main implementation, Logic Tensor Networks. It then traces how NeSy attaches to concrete BPM analytics tasks through a sequence of case studies spanning compliance-aware prediction, robust rule injection, healthcare outcome prediction, and prescriptive analytics. The tutorial highlights the *failure modes* that practitioners must avoid when injecting domain knowledge into deep process models, examines how to evaluate whether injected knowledge actually helps, and closes with open problems and a practical decision aid for when NeSy is worth adopting.

Keywords: Neuro-symbolic AI · Business process management · Predictive process monitoring · Prescriptive process analytics · Domain knowledge injection

1 Why BPM Meets Neuro-Symbolic AI

Modern process analytics rests on two intellectual traditions that rarely meet. The *data-driven* tradition learns predictors directly from event logs: given a running case, a neural model forecasts the next activity, the remaining time, the case outcome, or an entire suffix. The *knowledge-driven* tradition, which has long been central to BPM, encodes what we already know about a process (DECLARE constraints, temporal-logic rules, service-level agreements, regulatory obligations, and reference process models) and reasons over it symbolically. Each tradition is strong exactly where the other is weak. Deep models generalize from data but ignore explicit knowledge and need many

examples; symbolic models carry knowledge but degrade on the noisy, incomplete, and unlabeled data that real logs supply.

The cost of keeping the two apart is concrete and familiar to BPM practitioners. A purely neural outcome predictor can produce forecasts that violate constraints the organization *knows* must hold, such as predicting an activity that cannot legally precede another, or a decision that breaches a documented business rule. The problem is sharpest precisely where the stakes are highest and compliant behavior is rare. In rule-poor regimes a data-only model has almost nothing to imitate and collapses on rule adherence, even while its average accuracy looks acceptable.

Neuro-symbolic AI is the integration layer that lets a single model learn from data *and* respect knowledge. The field is young but maturing rapidly, and the BPM community has produced a distinct and growing cluster of NeSy work over the last two years. The aim of this tutorial is to make that work accessible to a general BPM audience: to explain what neuro-symbolic AI is, where it fits into the BPM analytics pipeline, what it provides, and when and how it fails. We assume familiarity with event logs and outcome prediction, and with the basics of deep learning, but *no* prior exposure to neuro-symbolic AI, fuzzy logic, or logic tensor networks [3,4].

2 Foundations of Neuro-Symbolic AI for BPM

We organize the landscape with Kautz’s six-type architectural taxonomy of neuro-symbolic systems [3], which ranges from loose pipelines (a symbolic component feeding a neural one, or vice versa) to tight couplings in which logic is compiled into the network itself. Most current BPM work occupies the *Type-4* slot, in which symbolic knowledge is compiled into the loss function of a neural learner. A recurring caveat, which we emphasize throughout, is that Type-4 integration delivers *training-time soft* compliance (the network is *encouraged* to satisfy constraints) rather than the runtime *hard* guarantees that some safety-critical settings demand. Marra et al.’s survey [4] supplies a complementary vocabulary, casting neuro-symbolic systems along shared dimensions (how knowledge is represented, how it is combined with learning, how it is used at inference) that let participants compare BPM methods against the wider AI literature.

Learning from Logical Constraints (LfLC). The paradigm underlying most Type-4 BPM work treats knowledge injection as constrained optimization. Instead of training only against data, the model minimizes a combined objective

$$\min_{\theta} \mathcal{L}_{\text{data}}(\theta) + \lambda \cdot \mathcal{L}_{\text{logic}}(\theta),$$

where $\mathcal{L}_{\text{logic}}$ penalizes violations of domain rules and λ controls how strongly knowledge is enforced relative to fit. Building this objective takes three conceptual steps: predicates of interest (e.g. “this prefix is compliant”, “lactate is high”) become learnable neural functions; crisp logical connectives are relaxed into differentiable operators via fuzzy t-norms; and the resulting truth values are aggregated into a loss term that backpropagation can optimize.

Logic Tensor Networks (LTN). LTN [1] is the main implementation of LfLC and underlies the BPM case studies in this tutorial. Through its *Real Logic* formalism, first-order axioms are grounded onto tensors: constants and variables become real vectors, predicates become neural networks returning truth values in $[0,1]$, and quantifiers become differentiable aggregations. This lets a practitioner write a domain rule as a first-order axiom and have it compiled into a training signal for a deep learning model. We give participants just enough Real Logic to read and write such axioms, without assuming any prior logic background.

3 Neuro-Symbolic AI Across Predictive and Prescriptive Process Analytics

The core of the tutorial walks through how neuro-symbolic AI attaches to concrete BPM analytics tasks. Rather than a single technique, the case studies each show a different point of contact between domain knowledge and a deep process model, and each surfaces a different lesson, including that knowledge injection can backfire.

Compliance-aware predictive monitoring: where does each rule attach? The first case study [7] addresses the most direct question a practitioner faces: given a rule and a predictive model, *where* should the rule be injected? It introduces a three-way injection scheme routed by the logical form of the rule: (A) *Feature Expansion*, augmenting the input with rule-derived signals; (B) *Output Refinement*, shaping the prediction head; and (C) *Parallel Constraints*, adding a dedicated constraint branch. Output Refinement gives the strongest empirical results, with F1 improvements of up to +18.4% on Sepsis and +10.1% on BPIC2012, and compliance that holds even when only 4.55% of training traces are compliant. This illustrates the data-efficiency argument for NeSy: knowledge substitutes for the labeled examples a pure-neural model lacks.

When knowledge hurts: vacuous satisfaction and rule pruning. The second case study [9] is the cautionary counterpart and is central to the tutorial’s message. Naively adding a logic loss can make a model *worse* than using no knowledge at all: on Sepsis, an unguarded LTN drops F1 to 18.79%. The cause is *vacuous satisfaction*: the model learns to satisfy an implication by making its premise false rather than by learning the intended behavior, collapsing onto a degenerate solution that is logically “correct” yet useless. The remedy is a two-stage training procedure: a weighted pretraining phase that prioritizes data fit, followed by analytical rule pruning that gates each axiom by its satisfaction dynamics ($g_i = \bar{s}_i \cdot e^{-\lambda \text{Var}(s_i)}$), retaining only rules that are both consistently satisfiable and informative, and a fine-tuning phase. This recovers performance to 75.82%. The takeaway is general: knowledge injection is not inherently beneficial; the mechanism that injects it must be designed with its failure modes in mind.

Outcome prediction in healthcare. The third case study [8] transfers the pattern to a high-stakes time-series setting by reading an ICU stay as a process: each patient stay is a trace of clinical events, and mortality is the case outcome. Domain rules are extracted automatically from SNOMED CT via AnyBURL, and predicates are

weakly anchored at clinically meaningful thresholds (e.g. `HighLactate` initialized at >4.0 mmol/L) so that learning starts from clinical knowledge but can adapt. The model offers *three-tier explainability*, at the level of concepts, rules, and individual patients, which makes it usable for real-time decision support and yields materially fewer false-positive alerts at a fixed recall.

Beyond prediction: prescriptive analytics and design choices. The final part broadens from prediction to *prescription* and uses contrasts to teach design judgment. NESYPA [6] brings neuro-symbolic reasoning to prescriptive process analytics (recommending interventions rather than forecasting outcomes) and deliberately steps *outside* the LILC paradigm, which makes it a useful boundary case for when logic-as-loss is the wrong tool. Recent external work on suffix prediction [5] shows that logic-as-loss *can* enforce sequential, temporal constraints if fuzzy aggregation is replaced by differentiable automaton execution (a DeepDFA-style temporal-logic loss); this is a useful teaching point about why some constraints resist naive fuzzy injection. We close with a data-efficiency result from neuro-symbolic anomaly detection [2], where measurable gains arise from as few as ten conformant traces, reinforcing that knowledge is most valuable when data is scarce.

4 Open Problems and Takeaways

The case studies form a consistent picture and expose questions that remain open.

Where the knowledge comes from. Every case study assumes a usable set of rules, and their quality bounds what NeSy can achieve. In practice rules must be elicited or extracted, as in the automatic extraction from SNOMED CT via AnyBURL in the healthcare study. Scaling this step, detecting conflicts among rules, and weighting each rule by its confidence rather than treating all axioms as equally hard are largely unsolved.

The soft-compliance gap. Type-4 integration buys compliance during training, not guarantees at inference. A model trained with a logic loss is more compliant on average, but nothing prevents it from violating a constraint on a particular case. Closing this gap by combining training-time logic losses with inference-time enforcement (constrained decoding, shielding, or post-hoc repair of predictions) is a pressing open problem for safety-critical BPM deployments.

Sequential constraints and the limits of fuzzy logic. The suffix-prediction contrast shows that fuzzy aggregation handles atemporal constraints well but struggles with the ordering and temporal-logic properties that are central to BPM. Differentiable automaton execution is one answer; how far fuzzy, LTN-style methods can be pushed before a different mechanism is required, and how to choose between them for a given constraint, is an open methodological question that the rule-pruning failure mode makes concrete.

Evaluating the utility of injected knowledge. A subtler issue underlies every reported gain: injected knowledge has value only insofar as it offsets the data signal, so a sound evaluation must isolate what knowledge adds *beyond* what the data already provides. Two conditions can make that contribution illusory. First, if the constraint is already implicit in the training data, for instance when most traces are compliant, the network learns it from data alone and the logic term is redundant; the rule-poor regimes that recur in the case studies (only 4.55% compliant traces in Sepsis and 13.09% in BPIC2012) are precisely the conditions under which knowledge can demonstrably help. Second, if the test data does not exercise the constraint, no metric can register the knowledge’s effect, whereas a test set skewed toward constraint-relevant cases can inflate it, so the test distribution decides whether utility is even observable.

A decision aid: is NeSy worth it for your problem? The tutorial distills the case studies into five diagnostic questions practitioners can apply to their own work: (1) is labeled data abundant or scarce; (2) are reliable domain rules available; (3) what is the regulatory exposure of a wrong or non-compliant prediction; (4) how much model opacity can stakeholders tolerate; and (5) what is the deployment horizon. NeSy pays off most clearly when data is scarce, rules are available, and compliance or explainability is non-negotiable, exactly the regime of the compliance-aware and healthcare studies. When data is abundant and rules are thin, a well-tuned neural baseline is often the better engineering choice, and the rule-pruning study is a reminder that injecting knowledge carelessly can do real harm.

References

1. Badreddine, S., d’Avila Garcez, A.S., Serafini, L., Spranger, M.: Logic tensor networks. *Artif. Intell.* **303**, 103649 (2022)
2. Gaikwad, D., van der Aalst, W.M.P., Park, G.: Neuro-symbolic process anomaly detection. In: Fuentes, L., Plebani, P., Combi, C., Reijers, H.A. (eds.) *CAiSE 2016*. LNCS, vol. 16559, pp. 240–255. Springer (2026)
3. Kautz, H.A.: The third ai summer: Aai robert s. engelmore memorial lecture. *AI Magazine* **43**(1), 105–125 (2022)
4. Marra, G., Dumancic, S., Manhaeve, R., Raedt, L.D.: From statistical relational to neurosymbolic artificial intelligence: A survey. *Artif. Intell.* **328**, 104062 (2024)
5. Mezini, A., Umili, E., Donadello, I., Maggi, F.M., Mancanelli, M., Patrizi, F.: Neuro-symbolic predictive process monitoring. *CoRR* **abs/2509.00834** (2025)
6. Park, G.: Neuro-symbolic prescriptive process analytics: Learning compliant and effective process recommendations through constraint distillation. In: Gal, A., Maggi, F.M., Leopold, H., Santoro, F. (eds.) *BPM 2026*. LNCS, Springer (2026)
7. Santis, F.D., Park, G., van der Aalst, W.M.P., Zanichelli, F.: Compliance-aware predictive process monitoring: A neuro-symbolic approach. In: Fuentes, L., Plebani, P., Combi, C., Reijers, H.A. (eds.) *CAiSE 2016*. LNCS, vol. 16559, pp. 223–239. Springer (2026)
8. Santis, F.D., Park, G., Zanichelli, F.: Neuro-symbolic artificial intelligence for real-time sepsis mortality prediction. *Eng. Appl. Artif. Intell.* **177**, 114920 (2026)
9. Santis, F.D., Park, G., Zanichelli, F.: Neuro-symbolic learning for predictive process monitoring via two-stage logic tensor networks with rule pruning. In: Wong, R.C., Tong, H., Lu, H., Kwok, J.T., Salim, F.D., Song, Y., Yiu, M.L. (eds.) *PAKDD 2026*. LNCS, vol. 16598, pp. 104–118. Springer (2026)