Research Diary

DPhil: Computer Science

Author: **Annabel Jakob**

Start Date: 24 February 2026

Field: **Computer Science**

Contents

1 Resources

- DPhil Progression Information and Resources

2 04 February 2026

2.1 Research Plan

Today's main tasks:

- Meeting with Seth and Gunes
- Familiarise with DPhil milestones and progression requirements
- Reading relevant literature suggested during supervisor meeting

2.2 Content Details

Meeting Summary

Meeting Notes:

- DPhil Milestones
  - Term 4, Week 0: Transfer status
  - 8th/9th Term: Confirmation of DPhil status
- Finding a research question
  - Bayesian Transformers
    - The Bayesian Geometry of Transformer Attention
    - Transformers Can Do Bayesian Inference
  - AI-assisted proofs
    - Existing tools: LEAN Proof Checker and Xena Project, Autodiff
    - Possible steps:
      1. Compile dataset of theorems presented in previous conference papers (e.g. NeurIPS)
      2. Verify the theorems and proofs in the dataset
      3. Goal: Can we produce *new* theorems and proofs?
    - Other resources:
      - `https://en.wikipedia.org/wiki/Kevin_Buzzard`
      - `https://terrytao.wordpress.com/2025/12/08/the-story-of-erdos-problem-126/`
      - Aristotle: IMO-level Automated Theorem Proving
- Mechanistic interpretability

  - Potentially connected to encrypted backdoors
  - Talk to Marek and Junayed
- Statistical Machine Learning
  - DeepRV: Accelerating spatiotemporal inference with pre-trained neural priors
- Random Number Generators
  - Can a backdoor be hidden in a RNG/ induced through an RNG? I.e. malicious signal induced via carefully chosen "random" numbers?
  - Planting Undetectable Backdoors in Machine Learning Models
  - Oblivious Defense in ML Models: Backdoor Removal without Detection

3 09 February 2026

Today's main tasks:

- Read Bayesian Transformers papers:
  - The Bayesian Geometry of Transformer Attention
  - Transformers Can Do Bayesian Inference

3.1 Paper - Transformers Can Do Bayesian Inference

Paper Reading

Paper title: Transformers Can Do Bayesian Inference [1]
Authors: Mueller et al.
Summary: Bayesian methods are usually slow or mathematically intractable for large datasets. Deep learning is fast but often bad at uncertainty and priors. As a solution, the authors present Prior-Data Fitted Networks (PFNs), which learn the mapping of the data to Bayesian posterior prediction.

Key Contributions

- PFNs (transformers that learn to approximate Bayesian posterior predictive distributions); this includes architectural changes to the standard transformer architecture, such as the Riemann distribution (a discretized output distribution for regression) and removal of positional encodings for permutation invariance over dataset examples

- Demonstrate that PFNs can approximate the PPD of Gaussian processes and Bayesian neural networks (BNNs) + are much faster than standard methods for approximating Bayesian inference (e.g. MCMC, variational inference)
- A BNN prior over architectures outperforms XGBoost/CatBoost on small tabular datasets while being ∼5000× faster, with strong calibration (low expected calibration error)
- Strong few-shot learning on Omniglot using a simple stroke-based prior

---

Notes

- Focuses on Bayesian posterior prediction for supervised learning problems, particularly small-data regimes (e.g. 30 training samples in tabular experiments)
- PFNs: train transformers to approximate Bayesian posterior predictive distributions by sampling synthetic datasets from a prior, masking labels, and learning to predict them. At inference time, a single forward pass approximates Bayesian inference.
- PFNs work for any prior distribution that can be sampled from → this is a very relaxed requirement compared to the standard assumptions of other Bayesian inference approximations. E.g. MCMC methods like NUTS [2] assume access to non-normalised posterior density, ability to evaluate the likelihood $p(D|\theta)$ and prior $p(\theta)$ pointwise, and gradients to the log-posterior.
- Deep learning models encode inductive bias through e.g. architecture, training data, and training procedure. PFNs let you inject an explicit prior by defining how to sample synthetic tasks.
- Limitation: PFN must be retrained for each new prior

Thoughts

- PFNs can effectively mimic Gaussian processes, but what about non-Gaussian processes (especially compared to MCMC and VI)?
  - The paper does test beyond GPs—the BNN experiments involve highly non-Gaussian, multimodal posteriors. PFNs handle these well precisely because they only need to sample from the prior, not evaluate densities. The limitation is more about whether you can design a prior that matches your real-world problem.

4 10th February 2026

Daily Summary

Today's main tasks:

- Meeting with Seth

4.1 Supervisor Meeting

Meeting Summary

Meeting Notes:

- Start working on first project in the next 2-4 weeks. Look at conference dates in the next 6 months and work towards a paper submission deadline.
- What does a DPhil thesis look like?
  - → It's essentially like writing + submitting a Master's thesis on a different topic
  - → Look at Seth's thesis (although it was essentially an integrated thesis)
- Talk to Dan Jensen about Bayesian wind tunnels and small recursive models

5 11th February 2026

Daily Summary

Today's main tasks:

- Read Bayesian Geometry of Transformer Attention [3]

5.1 Paper - The Bayesian Geometry of Transformer Attention

Paper Reading

Paper title: The Bayesian Geometry of Transformer Attention [3]
Authors: Agarwal et al.
Summary: Uses *Bayesian wind tunnels* (controlled prediction tasks with closed-form posteriors where the hypothesis space is too large for memorisation and in-context prediction requires genuine probabilistic inference) to empirically prove transformers perform exact Bayesian inference and understand how.

Key Contributions

- Transformers match analytic Bayesian posteriors with $10^{-3} - 10^{-4}$ bit accuracy on bijection learning and HMM filtering.
- Decompose Bayesian Learning into three primitives:
  - *Belief accumulation* integrating evidence into the posterior as observations arrive
  - *Belief transport* propagating beliefs forward through stochastic dynamics; i.e. transforming the belief vector according to the system dynamics.
  - *Random-access binding* Retrieving stored hypotheses by content rather than position.
- Architecture comparison: Transformers realise all three; Mamba handles accumulation/transport but struggles with binding; LSTMs only handle accumulation; MLPs fail entirely

Notes

- The authors used the different wind tunnels to identify which inference primitive the model realised. Belief accumulation was tested by bijection elimination, belief transport was tested by HMM filtering, and random-access binding was tested by associative recall.
- Transformers implement a three-step mechanism
  1. **Layer 0 - Set up representational infrastructure**: Build orthogonal frame (define hypothesis space). Keys at this layer form an approximately orthogonal basis over input tokens. Thus, each hypothesis occupies its own independent direction.
  2. **Middle layers - Perform actual computation**: Sharpen Q-K alignment (eliminate inconsistent hypotheses)
  3. **Late layers - Improve numerical accuracy**: Refine value manifold (encode posterior entropy precisely)

Thoughts

- The authors present three inference primitive, but they do not prove that these are *exhaustive*. Different tasks may reveal additional primitives.
- Key takeaways:
  - Wind tunnel idea: create tasks where you know the true Bayesian answer and memorisation is impossible. This lets us test whether models do genuine inference.
  - Transformers can do exact Bayes: On the wind tunnel tasks, transformers match the analytic posterior with high precision.
  - Architectures differ in what inference they can perform: The primitives framework gives us a vocabulary for why transformers beat LSTMs on some tasks (because attention enables content-based retrieval that recurrence cannot).
  - Attention creates interpretable geometric structure: Keys become orthogonal hypothesis axes, attention sharpens onto valid hypotheses across depth.

6 16th February 2026

Daily Summary

Today's main tasks:

- Read Aristotle: IMO-level Automated Theorem Proving [4]

6.1 Paper - Aristotle: IMO-level Automated Theorem Proving

Paper Reading

Paper title:	Aristotle: IMO-level Automated Theorem Proving [4]
Authors:	Achim et al.
Summary:	Aristotle is a system for automated theorem proving that can solve International Mathematical Olympiad (IMO) level problems. It uses a combination of formal verification and informal reasoning, which allows for flexible planning and reasoning and grants access to a much large corpus of mathematical proofs.

Key Contributions

- Aristotle: system for automated theorem proving that achieves gold-medal

performance on 2025 IMO problems

Notes

- Aristotle combines formal verification with informal reasoning
  - This allows it to leverage the large corpus of informal mathematical proofs + more flexibility in reasoning and planning (compared to fully formal systems like LEAN)
- Aristotle is composed of three main components:
  - *Lean proof search algorithm:* Core engine that builds proofs step-by-step using Monte Carlo Graph Search (MCGS)
    - A model trained using reinforcement learning serves two purposed during search: 1. Suggest which tactics to try from a given proof state and 2. estimate how likely a state is to be provable. This guides the search by predicting which proof steps are the most promising.
  - *Informal reasoning system:* Uses a language model to generate an informal proof sketch and breaks this proof down into lemmas which are then auto-formalized into LEAN.
  - *Geometry solver:* Specialized C++ engine for solving plane geometry problems.
- Aristotle workflow:
  1. Generate informal proof for target theorem
  2. Break informal proof into lemmas
  3. Auto-formalize lemmas into LEAN
  4. Proof search using MCGS
  5. If the proof search fails, revise the lemmas and repeat from step 3 until success or other termination condition.

Follow-up Reading

- Xena Project
- Monte Carlo Graph Search (MCGS)/ Monte Carlo Tree Search (MCTS)
- LEAN Proof Assistant
- AlphaZero and MuZero

7 17th February 2026

Daily Summary

Today's main tasks:

- Read DeepRV: Accelerating spatiotemporal inference with pre-trained neural priors [5]
- Create slides for supervisor meeting

7.1 Paper - DeepRV: Accelerating spatiotemporal inference with pre-trained neural priors

Paper Reading

Paper title: DeepRV: Accelerating spatiotemporal inference with pre-trained neural priors [5] authors

Authors:

Summary:

Key Contributions

- Novel decoder-only architecture: Eliminates the VAE encoder entirely, removing posterior collapse and oversmoothing errors that plague prior surrogates like PriorCVAE.
- Three architecture variants: MLP (baseline), gMLP (best accuracy/ efficiency trade-off), and Transformer (handles variable-length location sets via kernel-based attention).
- State-of-the-art fidelity: Outperforms INLA, inducing points, RFFs, ADVI, and PriorCVAE in matching full GP MCMC, on both predictive and hyperparameter recovery metrics.
- Flexible kernel support: Handles non-separable spatiotemporal kernals that INLA and RFFs cannot natively accommmodate.
- Practical scalability: Demonstrated on real-world city-scale dataset ($n = 4,994$ London LSOAs), completing in approx. 3 hours versus approx. 70 hours for full GP, on a single consumer GPU.

Notes

- **Problem:** Gaussian Processes (GPs) are the gold standard for spatiotemporal modelling but scale as $\mathcal{O}(N^3)$, making them intractable for large datasets. Existing approximations (INLA, sparce/inducing-point GPs,

RFFs, VAE-based surrogates like PriorCVAE) trade accuracy for scalability.

- **Core idea:** Decoder-only neural surrogate exploiting the GP Cholesky decomposition $\mathbf{f} = L\mathbf{z}$, learning the mapping $(\theta, \mathbf{z}) \to \hat{f}$ directly. This eliminates the VAE encoder, thus removing posterior collapse and oversmoothing errors.
- **Drop-in replacement:** Plugs into MCMC pipelines in place of GP prior sampling and therefore reduces complexity to $\prime(M^2)$.
- **Results:** Outperforms INLA, inducing points, RFFs, ADVI, and PriorCVAE on predictive MSE and hyperparameter recovery (Wasserstein distance to GP posteriors)
- **Flexibility:** Handles non-separable spatiotemporal kernels unsupported by INLA/RFFs.
- **Real-world:** London LSOA ($n = 4,994$) completed in approx. 3 hours vs. approx. 70 hours for full GP, on a single consumer GPU.
- Limitations: Pre-training data generation still requires $\mathcal{O}(N^3)$ Cholesky decompositions; gMLP requires fixed grid size at training time.

Keywords
- Gaussian Processes
- Kernel functions

8 18th February 2026

8.1 Current Status

Thoughts so far

- Found the Bayesian Transfomer [1, 3] and DeepRV [5] papers most promising.
- Aristotle [4] was interesting, but I didn't immediately have an intuition for possible extensions and the space seems somewhat saturated (the paper mentions another system developed independently at the same time as Aristotle which is very similar to their own).
- There should be a way to combine the Bayesian Transformer and DeepRV papers.

Reading Notes

Transformers Can Do Bayesian Inference [1]

- Propose *Prior-Data Fitted Networks* (PFNs)

→ framework for amortising Bayesian Inference via in-context learning

→ Transformer is meta-trained on dataset sampled from prior and learns to approximate the posterior predictive distribution in a single forward pass

The Bayesian Geometry of Transformer Attention [3]

- Uses Bayesian wind tunnels (tasks with known analytic posteriors and where memorisation is impossible) to empirically show that small transformers achieve near-exact Bayesian posteriors
- Decomposes Bayesian inference into three primitives (belief accumulation, belief transport, random-access binding) and shows which architectures (Transformers, Mamba, LSTM, MLP) realises which primitives + geometric analysis of the internal mechanisms (orthogonal key bases, progressive QK sharpening, value manifolds)

DeepRV: Accelerating spatiotemporal inference with pre-trained neural priors [5]

- Proposes a neural surrogate for Gaussian Processes in spatiotemporal inference using a decoder-only architecture that directly learns the Cholesky factor mapping.

8.1.1 Project Ideas

- The Bayesian Geometry paper defines three primitives that correspond to the tasks that they used → the list of primitives is not necessarily exhaustive and we could define additional primitives for tasks not included in the paper
- The Bayesian Geometry paper only tests standard transformers, Mamba, LSTMs, and MLPs. We could test additional architectures
- DeepRV replaces GP priors in MCMC pipelines but still requires MCMC for inference → could combine DeepRV and PFNs so that DeepRV handles the prior and PFNs replace MCMC

→ verify through wind tunnels

→ for PFN, the prior over functions would be defined by DeepRV

8.2 Supervisor Meeting

Meeting Summary

Paper Discussion

- Bayesian Geometry
  - How are the models trained? How are they evaluated?
  - What is the distribution over?
  - Güneş used to work with pyprop → big overlap with what the Bayesian

Geometry paper did
  - DeepRV without MCMC was already done. See BSA-TNP [6]
  - Talk to Dan Jensen about this
- PFNs
  - how do they get from a prior-data fitted network to the posterior?

Notes

- Look at fundamental computational Bayesian Theory:
  - Aki Vehtari homepage
  - Probability Theory: The Logic of Science
  - Bayesian Data Analysis
  - David J. C. MacKay Wiki Page
  - Information Theory, Inference, and Learning Algorithms
- Evaluate BSA-TNP and/or DeepRV using wind tunnel

→ Güneş has a lot of data for this (particle physics)

→ See Efficiently Sampling Functions from Gaussian Process Posteriors

9 20th February 2026

Daily Summary

Today's main tasks:

- Review BSPP course materials and other Bayesian inference resources suggested by supervisor meeting

10 21st February 2026

Daily Summary

Today's main tasks:

- Review BSPP lectures 3 and 4

- Re-read Transformers Can Do Bayesian Inference [1] with a focus on understanding the training and evaluation procedures in more detail

10.1 Paper: Transformers Can Do Bayesian Inference

Paper Reading

Paper title: Transformers Can Do Bayesian Inference [1]
Authors: Mueller et al.
Summary: Bayesian methods are usually slow or mathematically intractable for large datasets. Deep learning is fast but often bad at uncertainty and priors. As a solution, the authors present Prior-Data Fitted Networks (PFNs), which learn the mapping of the data to Bayesian posterior prediction.
Github repo: https://github.com/automl/TransformersCanDoBayesianInference

Questions to clarify

- How do they get from a prior-data fitted network to the posterior?

Notes

- Method for *posterior approximation* formulated as a supervised classification problem

→ i.e. the goal is to find a model that approximates the *posterior predictive distribution*

→ Authors consider a prior $p(t)$ where the latent variable t is the task.

- The training procedure is exactly the same as standard supervised learning. The key difference is that the training data is generated by sampling synthetic datasets from a prior distribution, masking the labels, and learning to predict them. Thus, the model learns to map from datasets to posterior predictions.

→ For each training step, samples a dataset $D = \{(x_i, y_i)\}_{i=1}^{n}$ (these are the **context tokens**) and query tokens x from the same prior distribution $p(\mathcal{D})$. The label of the query is *hidden* (zeroed out or set to a special mask token). The loss is then computed only for the query tokens.

→ Uses *Prior-Data Negative Log-Likelihood* (Prior-Data NLL):

$$\ell_\theta = \mathbb{E}_{D \cup \{x,y\} p(\tilde{\mathcal{D}})}[-\log q_\theta(y|x, D)]$$

where $D \cup \{x, y\}$ is a dataset of size $|D| + 1$ sampled from $p(\mathcal{D})$

→ ℓ_θ is equal to the expectation of the cross-entropy between the PDD and the approximation q_θ.

- **Meta-learning** / **Learning-to-lean**: Goal is to generalize learning meth-

ods from a training set of datasets to a validation set of datasets
- Architecture: Based on Transformer encoder without positional encoding (to ensure permutation invariance over dataset examples)
- Conducted the experiments using *Riemann Distributions*, i.e. discretized continuous distribution
- Experiments approximated Gaussian Processes and Bayesian Neural Networks
- Experiments on real-world datasets:
  - Tabular classification: Use a GP prior with hyper-priors (distribution over hyperparameters). To make the dataset a classification prior, the authors computed the median of all targets in the dataset and set the class to one if greater than the median and zero otherwise.
  - Use BNN to learn a prior over model architectures. Authors define the prior probability over the space of model architectures (see [1] for sampling procedure)

11 23rd February 2026

11.1 Paper: The Bayesian Geometry of Transformer Attention

Paper Reading

Paper title:
Authors:
Summary:
papertitle = The Bayesian Geometry of Transformer Attention [3], authors = Agarwal et al., summary = Uses *Bayesian wind tunnels* (controlled prediction tasks with closed-form posteriors where the hypothesis space is too large for memorisation and in-context prediction requires genuine probabilistic inference) to empirically prove transformers perform exact Bayesian inference and understand how.

Questions to clarify

- How are the models trained? How are they evaluated?
- What is the distribution over?

Notes

- Setup
  - Given a family of tasks indexed by $\theta\pi(\tilde{\theta})$, inputs x are drawn from

some distribution, labels are drawn according to $yp(y|\tilde{x}, \theta)$. The model observes context $c = \{(x_i, y_i)\}_{i=1}^{k}$ and must predict y for a new query input.
  - Loss function is the population cross-entropy, just as in [1] (here, the authors sample a new task for each batch though)
- Evaluation
  - "A model that achieves the correct posterior entropy at every position is functionally Bayesian - it produces predictions with the same uncertainty profile as the exact posterior"
  - In general, entropy matching doesn't guarantee distributional equivalence since two very different distributions can have the same entropy
  - → just measuring entropy MAE doesn't tell us if the model had learned the right distribution, only the right uncertainty level.
  - → The authors then argue that in their specific tasks, entropy does act as a sufficient diagnostic because the constrained structure of the task severely limits which distributions are consistent with a given entropy value. In addition, they verify distribution matching using KL divergence
  - → The main metric used is the *entropy calibration error*:

$$MAE = \frac{1}{L} \sum_{k} |H_{\text{model}}(k) - H_{\text{Bayes}}(k)|$$

- How Transformers perform exact Bayesian inference
  - They first construct a representational frame, execute sequential elimination within this frame, and then refine posterior precision across layers.

Thoughts

- Authors use small transformers. Do the findings translate to larger models?

References

[1] Samuel Müller et al. "TRANSFORMERS CAN DO BAYESIAN INFERENCE". In: *ICLR 2022 - 10th International Conference on Learning Representations*. 2022.

[2] Matthew D. Hoffman and Andrew Gelman. "The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo". In: *Journal of Machine Learning Research* 15 (2014). ISSN: 15337928.

[3] Naman Agarwal and Siddhartha R Dalal. "The Bayesian Geometry of Transformer Attention Paper I of the Bayesian Attention Trilogy". In: 1 (2026). URL: https://arxiv.org/abs/2512.22471.

[4] Tudor Achim et al. *Aristotle: IMO-level Automated Theorem Proving*. 2025. arXiv: 2510.01346 [cs.AI]. URL: https://arxiv.org/abs/2510.01346.

[5] Jhonathan Navott et al. "DeepRV: Accelerating spatiotemporal inference with pre-trained neural priors". In: (Mar. 2025). URL: https://arxiv.org/abs/2503.21473v2.

[6] Daniel Jenson et al. *Scalable Spatiotemporal Inference with Biased Scan Attention Transformer Neural Processes*. Tech. rep. 2025.