Few-Shot Open-Set Classification via Reasoning-Aware Decomposition

Avyav Kumar Singh, Helen Yannakoudakis
Department of Informatics
King's College London
{avyav_kumar.singh, helen.yannakoudakis}@kcl.ac.uk

Abstract

Large language models (LLMs) excel at few-shot learning, but their ability to reject out-of-distribution examples remains under-explored. We study this challenge under the setting of *few-shot open-set classification*, where a model must not only classify examples from a small set of seen classes but also reject unseen ones at inference time. This setting is more realistic and challenging than traditional closed-set supervised learning, requiring both fine-grained classification and robust rejection. We show that, for small LLMs, neither chain-of-thought (CoT) prompting nor supervised fine-tuning (SFT) alone are sufficient to generalise reliably, particularly when class semantics are anonymised. We introduce Wasserstein GFN (W-GFN), a novel amortised Generative Flow Network framework that uses latent trajectories to approximate the Bayesian posterior. With as few as 4 examples per class, W-GFN substantially improves performance, enabling Llama 3.2 3B to achieve up to $\geq 80\%$ of the performance of Llama 3.3 70B in complex datasets, despite being ~ 23 times smaller, which highlights the importance of reasoning-aware approaches for robust open-set few-shot learning.

1 Introduction

Generative large language models (LLMs) have been shown to excel in few-shot learning (Brown et al., 2020; Gao et al., 2021; Zhao et al., 2021), where tasks are performed using only a handful of labelled examples at inference time. This is typically achieved through in-context learning, where in the test setting the classes are considered *seen*, since examples from the same classes as the test instances are provided as demonstrations. However, the mechanisms driving this generalisation remain under-explored – particularly the distinction between genuine reasoning about input data versus reliance on surface cues or shallow memorisation of patterns from pretraining.

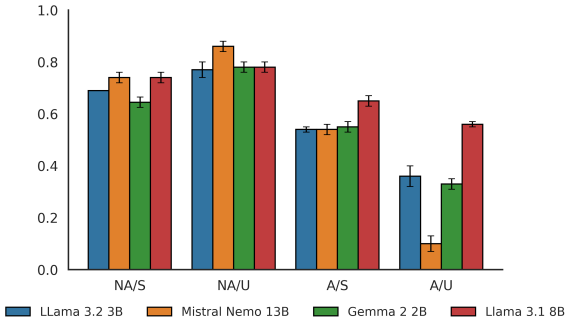


Figure 1: Few-shot CoT F1-score for **S**een / **U**nseen classes with **N**on-**A**nonymised and **A**nonymised labels.

This question becomes especially pressing in *few-shot open-set classification* (OSC), where models must classify among seen demonstration classes while rejecting examples from unseen ones – unlike traditional supervised learning, which assumes a shared label space for training and testing. OSC is a more realistic and challenging setting, requiring to handle both known in-distribution (IID) and unknown/unseen out-of-distribution (OOD) samples at test time. Under the generalised OOD detection framework (Yang et al., 2024), OSC involves test-time semantic shift: $P_{train}(Y) \neq P_{test}(Y')$. Few-shot OSC is particularly challenging for smaller LLMs, which, in the absence of rich prompts, may fall back on surface cues or memorised associations from pretraining rather than demonstrating genuine generalisation. While more accessible and efficient for constrained environments, these models struggle more with generalisation. Nonetheless, studying few-shot OSC in smaller LLMs remains important, as it reflects real-world needs where both efficiency and the ability to distinguish known from unknown classes are critical (Liu et al., 2020; Geng et al., 2021).

Motivation To better understand the challenges of OSC, we examine the use of chain-of-thought (CoT) prompting on the MultiNERD named entity

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 13854–13875
November 4-9, 2025

recognition dataset (Tedeschi and Navigli, 2022). From the training split, we sample 8 examples each from 4 random classes for few-shot prompting to encourage explicit reasoning. For evaluation, we sample 50 instances each from 10 random test classes, repeating this 3 times to create distinct test sets. We report F1-scores for seen classes (classified into demonstration classes) and unseen classes (correctly rejected as out-of-set). We compare two settings: (1) non-anonymised labels (e.g., *Person, Animal*), and (2) anonymised labels (e.g., *A, B*). The anonymised condition prevents the model from exploiting prior semantic knowledge of labels, forcing it to rely on reasoning over the demonstrations. By contrasting the two, we test whether models genuinely reason over inputs or instead rely on surface cues or memorised associations – a distinction that is critical for separating reasoning failures from memorisation. In both, the model chooses a class or *None of these*. Furthermore, predicting *any* non-demonstrated class for unseen instances (e.g., *Class N* or *Class Location*) is still regarded as correct rejection.[1]

In Figure 1, we evaluate four comparatively small LLMs, ranging from 2B to 13B. All models perform well with non-anonymised labels – even on unseen classes – by extrapolating to semantically related outputs (see Footnote 1). Interestingly, under anonymised labels, performance drops substantially and particularly for unseen classes (see prompts in Appendix A.1). This suggests that smaller LLMs may overly depend on label semantics (semantic memorisation) when available rather than engaging in example-based reasoning.

To address this, we focus on anonymised labels,[2] and propose introducing latent variables Z representing intermediate reasoning steps conditioned on input X and predicting Y by marginalising over Z. We propose maximising the posterior $P(Y|X) \propto \sum_i P(Y|Z_iX)P(Z_i|X)$ instead of directly optimising $P(Y|X)$. This encourages reasoning via latent trajectories rather than shortcutting to Y. Such reasoning-aware decomposition is well aligned with cognitive and probabilistic decision-making frameworks. It also mitigates overfitting, especially when Y lacks semantic cues, by promoting abstraction (i.e., intermediate concepts) over memorisation.

[1]This is a lenient evaluation setting: any incorrect prediction to an out-of-set label is accepted as 'None of these', artificially inflating unseen F1. We also prompt the model that some examples may be out-of-set (Appendix A.1). Thus, these results represent an upper bound on unseen performance.

[2]Anonymised labels are a common practice in prior work (e.g., encoder-based classification and meta-learning; Liu et al. (2020); Snell et al. (2017); Bansal et al. (2020a)).

2 Related Work

Few-shot learning There is a large body of research on few-shot learning with LLMs (Ji et al., 2023; He et al., 2024; Liu et al., 2024b; Singh et al., 2024; Yu et al., 2023). Existing approaches span a range of strategies, including learning with embeddings from (non-generative) language models (Snell et al., 2017; Bansal et al., 2020b; J. Reddi et al., 2023; Viswanathan et al., 2024), using few-shot demonstrations (with or without explanations) (Brown et al., 2020; Gao et al., 2021; Lampinen et al., 2022), prompt engineering (Bohra et al., 2023; Kaneko et al., 2024), and retrieval-augmented methods (Izacard et al., 2023; Cao et al., 2021). However, most assume a closed-set setting, where the label space is shared between training tasks and test inputs. Meta-learning approaches additionally rely on large, diverse sets of few-shot tasks to generalise well, thus limiting applicability in data-scarce settings (Hospedales et al., 2022).

Out-of-distribution detection These methods rely on LLM robustness (Hendrycks et al., 2020) combined with statistical tools such as Mahalanobis distance (Xu et al., 2020; Podol'skii et al., 2021; Zhou et al., 2021), likelihood ratios (Zhang et al., 2025), logit similarity (Liu et al., 2024a), or few-shot demonstrations (Wang et al., 2024). However, these typically distinguish in-distribution from OOD data, lacking fine-grained classification within the in-distribution space and few-shot capabilities (Chen et al., 2024).

Few-shot open-set intent detection This body of work is the closest to our setting. Existing generative LLM approaches rely on very large models such as ChatGPT (Wang et al., 2024; Song et al., 2023) and are benchmarked exclusively on intent detection tasks (Casanueva et al., 2020; Larson et al., 2019). Our work differs in that it targets small, decoder-based LLMs and introduces a reasoning-aware latent-variable framework to enable both classification and rejection in a unified, amortised way. We also demonstrate that label semantics play a critical role in model generalisation, and show that anonymising labels exposes core weaknesses in LLM reasoning – issues that are largely unaddressed in prior intent detection litera-

ture. Nevertheless, we take relevant baselines such as CoT prompting and apply them to our work, while also using the CLINC150 intent detection dataset (Larson et al., 2019) in our evaluation setup.

Our contribution We present W-GFN, a reasoning-aware,[3] amortised[4] approach to few-shot open-set classification, using as few as 4 examples per class. This is particularly valuable for smaller LLMs in data-scarce settings, where both compute and supervision are limited. Our method introduces latent-variable decomposition to encourage reasoning via intermediate abstractions, mitigating overfitting to label semantics. To the best of our knowledge, this is the first amortised approach to few-shot OSC for (small) LLMs. We release our code to facilitate research in this area: https://github.com/avyavkumar/few-shot-open-set-classification.

3 Generative Flow Networks

Overview Generative Flow Networks (GFNs) (Bengio et al., 2023; Malkin et al., 2022; Hu et al., 2024) are probabilistic models that learn to sample from structured distributions over complex discrete spaces by modelling generation as a sequence of actions, similar to a Markov Decision Process. Unlike traditional models that define a direct distribution over outcomes, GFNs train a stochastic policy so the marginal probability $P_T(s)$ of a state s is proportional to a reward $R(s)$, enabling sampling from unnormalised distributions via an energy function $E(s) = -\log R(s)$. Unlike diffusion processes (Ho et al., 2020; Yang et al., 2023) or VAEs (Kingma and Welling, 2014; van den Oord et al., 2017), GFNs match reward signals rather than relying solely on positive examples, making them well-suited for tasks like molecular generation (Zhu et al., 2023; Roy et al., 2023), causal discovery (Deleu et al., 2023; Atanackovic et al., 2023), and combinatorial optimisation (Zhang et al., 2023; Kim et al., 2024).

Language modelling GFNs are appropriate for LLM-style sequence generation, as they allow explicit reward shaping over sequences. As previously demonstrated (Hu et al., 2024), GFNs model a policy function over the generation of

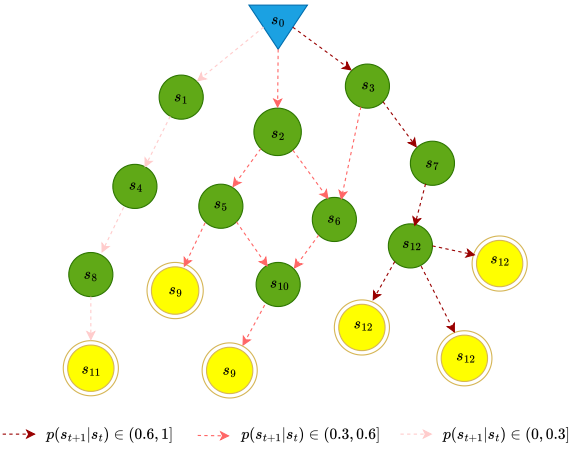


Figure 2: GFNs for language modelling. Blue state: input X; green: latent sequences Z; yellow: labels Y. Probability of generating label s_t is proportional to the unnormalised reward $R(XZY)$ where $Y = s_t$.

latent sequences $Z = z_1 z_2 \dots z_n \top \in \mathcal{Z}$, where $\top$ signifies a terminal stop token. The objective is to sample from a distribution over the space $\mathcal{Z}$, guided by an unnormalised reward function $R : \mathcal{Z} \to R_{>0}$. Starting from an empty string, a token z_i is sampled at each step according to the learned sampler $q_{\text{GFN}}(z_i \mid z_{1:i-1})$, and is appended to the partial sequence until $\top$ is drawn. The probability assigned to a complete sequence $Z = z_{1:n}\top$ by the policy is given by $q^{\top}_{\text{GFN}}(Z) = (\prod_{i=1}^{n} q_{\text{GFN}}(z_i \mid z_{1:i-1}))\, q_{\text{GFN}}(\top \mid z)$ – i.e., the trajectory-level probability from beginning to end – where the initial context z_0 corresponds to the empty string. The training objective of GFNs is to fit a parameterised sampler $q_{\text{GFN}}(\cdot \mid \cdot; \theta)$ such that the resulting distribution over terminal sequences satisfies $q_{\text{GFN}}(Z) \propto R(Z)$, aligning the sampling likelihood with the reward signal (see Figure 2).

Training a GFN To achieve the desired objective of a GFN – sampling a sequence of discrete states proportional to a reward signal R – we need to learn the forward transition policy $P_F(s_{t+1}|s_t)$, the backward transition probability $P_B(s_t|s_{t+1})$, and a flow function $F(s_t)$ with the following constraint: the total outgoing flow from state s_t must equal the total incoming flow at s_t (Bengio et al., 2023). The subtrajectory balance loss (Madan et al., 2023) enforces this between any pair of intermediate states $i \to j$ within a sampled sequence of tokens $Z = z_{1:n}\top$, and is written as:

$$\mathcal{L}_{SubTB}(Z) = \left(\log \frac{F(s_i) \prod_{t=i}^{j-1} P_F(s_{t+1}|s_t)}{F(s_j) \prod_{t=i}^{j-1} P_B(s_t|s_{t+1})} \right)^2$$

[3]Where the model operates over latent explanations / structured reasoning chains and scores those for alignment with downstream task performance.

[4]Where we train a single model, unlike approaches which use separate models to classify seen & detect unseen classes.

where F is parameterised by θ. For LLMs, at convergence, we have $R(s_n\top) = F(s_n)P_F(\top|s_n)$; therefore, Hu et al. (2024) define $F(s_n) = R(s_n\top)/P_F(\top|s_n)$. Trivially, $P_B(s_t|s_{t+1}) = 1$ for any fixed sampled sequence in natural language, as the next token deterministically follows the previous prefix. This allows us to simplify the loss for sampled sequences, and for all intermediate states $i \to j$ we have:

$$\mathcal{L}_{SubTB}(Z;\theta) = \sum_{1<j\leq n} \sum_{0\leq i<j} \left[\log \frac{R(z_{1:i}\top)\prod_{k=i+1}^{j} q(z_k|z_{1:k-1})q(\top|z_{1:j})}{R(z_{1:j}\top)q(\top|z_{1:i})}\right]^2 \quad (1)$$

where $q(\cdot|\cdot)$ refers to $q_{GFN}(\cdot|\cdot)$. This loss minimises the difference between the probability of transitioning forward from $i \to j$ and backward from $j \to i$ for each pair of token indices i and j in the sequence $R(z_{1:n}\top)$, thus satisfying the condition that incoming flows must equal outgoing flows. Details of other losses are in Appendix A.2.

4 Our method: Wasserstein-GFN

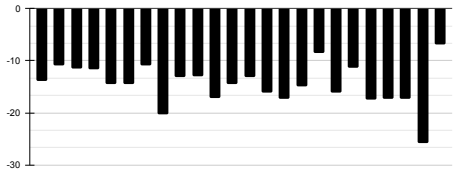
Training objective Following the episodic meta-learning definitions, given K data points for each of N classes, we construct an input *episode* X comprising a *support set* of few-shot demonstrations and a single *query* test example which can belong to any of the seen classes (see Figure 7, Appendix A.1). We generate multiple *training* episodes by sampling different class subsets and query instances and aim to correctly classify the query example using the provided few-shot demonstrations in the support set. On the other hand, the objective at *test time* is to correctly classify a query example into either the correct seen class or reject it as an out-of-distribution example.

Directly estimating posterior probabilities Hu et al. (2024) apply GFNs at the token level to generate latent sequences $Z \sim q_{GFN}(Z|X)$ and then rely on the pretrained (base) LLM to predict the label $Y \sim p_{LLM}(Y|XZ)$. This setup assumes that the base LLM can already assign high probability to the correct label Y given a latent Z – an assumption that often fails, particularly under anonymised labels – and they address this by fine-tuning the LLM on concatenated sequences XZY. By contrast, we reframe the role of GFNs entirely: rather than generating text tokens, we use GFNs as reasoning-aware posterior estimators that directly score latent reasoning paths with respect to a classification task. This allows us to approximate the marginal posterior $p(Y|X)$ without supervised fine-tuning of the base LLM, while still performing inference over latent reasoning chains provided by the LLM. Specifically, we estimate $p(Y|X) \propto \sum_Z p(Y|XZ)p(Z|X) \approx \sum_Z q_{GFN}(Y|XZ)p(Z|X)$, where the conditional prior latent $Z|X$ corresponds to latent sequences sampled from the base LLM conditioned on the input $X : Z \sim p_{LLM}(Z|X)$.

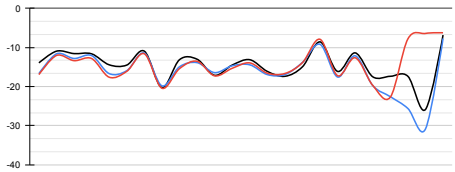
Sampling conditional prior latents Using the few-shot demonstrations, we sample prompt-based *support latents* (sequences obtained by prompting the base LLM to describe the class of the query example using only few-shot demonstrations), and *query latents* (sequences obtained by prompting the base LLM to describe only the query example) (see examples in Appendix A.3). We can then train q_{GFN} to maximise $\Sigma_Z p_{q_{GFN}}(Y|XZ)p_{LLM}(Z|X)$; however, to simplify training, we assume a uniform prior over sampled latents and directly maximise $\Sigma_Z p_{q_{GFN}}(Y|XZ)$ for all input-latent sequences XZ describing class Y.[5] To summarise, we collect conditional prior latent sequences $Z|X$ for query label Y and train q_{GFN} to map $Z \to Y$ by maximising $q_{GFN}(Y|XZ)\ \forall\ Z$. This aligns with classical graphical model formulations and enables amortised inference over structured latent spaces, allowing the model to reason over multiple latent explanations efficiently.

Logit normalisation trick Prior work (Hu et al., 2024) trains q_{GFN} towards higher reward trajectories by adding a negative offset Δ to undesirable trajectories (e.g., trajectories mapping a latent to an incorrect class in a classification task) – making low-reward trajectories less likely to be sampled under the GFN policy (Bengio et al., 2023). In our experiments, mapping latents to the correct class with a positive reward offset causes training instability, and $\mathcal{L}_{SubTB}(Z;\theta)$ in Equation 1 diverges (see Appendix A.4). Moreover, penalising all incorrect trajectories is both costly and ill-posed. Assigning negative rewards to all alternative labels incurs $O(NT)$ complexity per instance (N classes and T tokens in the sequence XZY), while the space of full reasoning paths is combinatorially

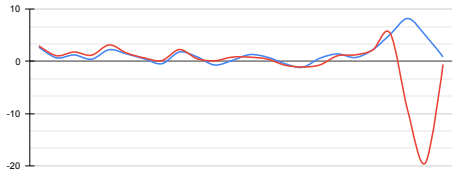
[5] This approximation is reasonable since sampling with lower temperatures yields high-probability priors.



(a) Base LLM log-probs for tokens in ZY ($p_{LLM}(ZY|X)$).



(b) Base LLM log-probs for $p_{LLM}(ZY|X)$ in **black**; token-wise log-probs for high- and low-probability sequences under $q_{GFN}(ZY|X)$ in **blue** and **red** respectively.



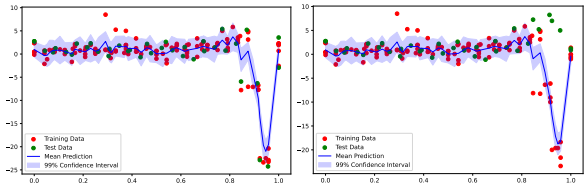
(c) In-distribution logit differences in **blue** and out-of-distribution logit differences in **red**.

Figure 3: By performing $p_{LLM}(XZY) - q_{GFN}(XZY)$ we are able to detect in-distribution and out-of-distribution latent $\rightarrow$ label mappings.

large. Furthermore, "incorrect" trajectories often lack a clear negative signal – many are partially plausible – so uniformly suppressing them risks penalising useful alternatives. To address these issues, we introduce a logit normalisation trick that is applied post-training: we normalise the token-wise log probabilities by subtracting them from the base model's token-wise log probabilities to get the relative score $\delta(XZY) = \log p_{LLM}(XZY) - \log q_{GFN}(XZY)$. This calibration ensures higher scores for correct latent$\rightarrow$label mappings (see Figure 3), acting as contrastive training without explicitly penalising alternatives.[6] It offers a scalable, stable objective that mitigates reward imbalance and reframes training in terms of relative confidence.

Wasserstein reward offset In Hu et al. (2024), for a latent sequence $Z_{1:t}$ of length T, the token-wise reward at step t is defined as $R(Z_{1:t}) = \prod p_{LLM}(XZ_{1:t}Y)$ with offset $\Delta < 0$ added at token t if $p_{LLM}(XZ_{1:t}Y_{correct}) < p_{LLM}(XZ_{1:t}Y_{incorrect})$. This penalises incorrect latent $\rightarrow$ label mappings with a low reward to discourage q_{GFN} from sampling these sequences. We

[6] Although this breaks autoregressive consistency of q_{GFN}, it is inconsequential here since we only compute posterior probabilities of fixed sequences.



(a) KL $(\mathcal{N}_0^A \,\|\, \mathcal{N}_1) = 17.44$ (b) KL $(\mathcal{N}_0^B \,\|\, \mathcal{N}_1) = 72.13$

Figure 4: GPs fit on OOD log probabilities (normalised by sequence length; X-axis) per class, with their respective KL divergences: OOD logits have low KL divergence (left); in-distribution ones have high (right).

propose using a gradually increasing negative offset on trajectories that yield the correct label (rather than a constant Δ), which leads to more stable training (see Section 6). Specifically, we add a negative reward offset Δ_t at token t to $R(Z_{1:t})$ using a Wasserstein interpolation between 0 and Δ as

$$\Delta_t \sim -\mathcal{N}((1-\alpha)\Delta_0 + \Delta \cdot \alpha, 1) \qquad (2)$$

where $\Delta_0 = 0, \alpha = t/T$. The offset is near zero at early steps but grows larger toward the end of the sequence. This reduces the relative dominance of correct paths without removing their overall advantage, preventing the GFN from collapsing too quickly onto a few reasoning chains. By narrowing the margin between correct and incorrect trajectories, the model is encouraged to explore a broader range of reasoning paths, which improves stability and yields a more informative reward signal without requiring dense supervision of negatives. Once the Wasserstein offset is applied to the correct sequences, we use the logit normalisation trick to compare token-wise log-probabilities between the base LLM and q_{GFN} and select high-confidence mappings from latents to correct labels.

Detecting high probability latent$\rightarrow$ label transitions automatically To identify whether a sampled latent sequence $Z \sim p_{LLM}(Z|X)$ leads to a high-confidence prediction for a class c, we analyse the distribution of token-wise log-probability differences between the base LLM and q_{GFN}, using the logit normalisation trick. For each class c, we generate negative training examples by using one query example X with a label that is not c (given by c^*) and compute their token-wise logit differences $\log p_{LLM}(XZY^{c^*}) - \log q_{GFN}(XZY^{c^*})\ \forall c^*$ – these sequences represent the characteristic out-of-distribution logit behaviour for c, and help us identify how far a new candidate trajectory diverges from what's not expected. We then fit a Gaussian

Process (Rasmussen, 2004) to these δ for class c using a linear mean and a Matérn kernel (with $\nu = 5/2$) (Genton, 2002). Given a test input X_{test}, we sample latent sequences Z and compute the token-wise logit differences. For each class c, we compare the test sequence's distribution of δ ($\mathcal{N}_1$) against the distribution modelled by the GP trained for c, f_c ($\mathcal{N}_0^c$). We then compute the closed-form KL divergence between these two distributions as

$$\mathrm{KL}(\mathcal{N}_0^c \,\|\, \mathcal{N}_1) = \frac{1}{2}\sum_{t=1}^{T}\Big[\log\frac{|\Sigma_1|}{|\Sigma_0|} - n + \mathrm{tr}(\Sigma_1^{-1}\Sigma_0) + (\mu_1^t - \mu_0^t)^\top \Sigma_1^{-1}(\mu_1^t - \mu_0^t)\Big]$$

where μ_0^t is the mean of f_c at token position t, μ_1^t is the logit δ at t for the test sequence, and Σ_0 and Σ_1 are the covariance matrices of f_c and the test example respectively. Note that any distribution of token-wise log probabilities that deviates substantially from these modelled low-confidence, out-of-distribution log-probabilities will be interpreted and classified as in-distribution for class c. Specifically, we apply Z-Score normalisation across the list of class-wise KL divergences. If the Z-score for a class c is $Z_c^s \geq 3.5$ (i.e., it reflects a strong statistical deviation), we classify the test example as c as this is a high probability latent $\rightarrow$ label mapping. If there is no such outlier detected for any of the classes, then we predict that the test example as out-of-distribution (see Figure 4).

Core principle of our approach Training q_{GFN} to map latent sequences to a label encourages it to explore nearby sequences and associate them with similar rewards for that label (Bengio et al., 2023; Hu et al., 2024), thereby inducing locally consistent reward neighbourhoods in latent space. This helps generalise beyond specific latent sequences and capture broader distributional structure associated with each label. We then classify a test example as either *seen* class or *unseen* by examining the distribution of logit differences, enabling robust label prediction and principled OOD detection using posterior probability signals. Our training and test process is shown in Algorithms 1 and 2.

Algorithm 1: Training Wasserstein-GFN

```
q_GFN ← parameterised by θ
{X, Y} ← N(K − 1) episodes and labels
{X*, Y*} ← K episodes and labels
L_SubTB(Z; θ) ← 0
for x, y ∈ {X, Y} do
    Z ∼ p_LLM(x), Z = {Z_1, Z_2...Z_m}
    for Z ∈ Z do
        for token t ∈ Z do
            R(Z_{1:t}) += Δ_t using Eq. 2
        end
        Calculate l_SubTB(Z; θ) using Eq. 1
        L_SubTB(Z; θ) += l_SubTB(Z; θ)
    end
    θ ← θ − ∇_θ L_SubTB(Z; θ)
end
for x^c, y^c ∈ {X*, Y*} do
    Z ∼ p_LLM(x^c), Z = {Z_1, Z_2...Z_m}
    diff ← {}
    for y ∈ {Y*} − y^c do
        diff.append(p_LLM(x^c Z y) − q_GFN(x^c Z y))
          ∀Z ∈ Z
    end
    Train f_c using diff
end
return q_GFN, {f_1, f_2...f_c}
```

Algorithm 2: Prediction with W-GFN

```
q_GFN ← parameterised by θ
{f_1 : f_c} ← GPs fitted on log-probs
{X} ← test example
{Y} ← all labels in support set
Sample Z = {Z_1|X, Z_2|X...Z_m|X}
preds ← []
for Z ∈ Z do
    d[c] ← [KL(δ(XZy_c)||f_c)] ∀ y_c ∈ Y
    Z^s ← Z-Score normalisation of d
    if Z^s[c] ≥ 3.5
        preds.append(y_c)
    else
        preds.append(out-of-distribution)
end
return max_count(preds)
```

5 Tasks and Baselines

Datasets We evaluate on three tasks of increasing complexity: NER (MultiNERD; Tedeschi and Navigli (2022)), intent detection (CLINC150; Larson et al. (2019)), and emotion classification (GoEmotion; Demszky et al. (2020)). MultiNERD serves as a structured, pattern-driven classification task; CLINC150 offers high-level domain-specific user utterance classification, at times with surface-level cues (e.g., highly specific domains such as *Credit Cards* contain the words *credit* or *card* in many of the train/test examples); GoEmotion presents a challenging fine-grained emotion classification task, requiring models to generalise across highly diverse linguistic expressions, while also handling annotation subjectivity, multi-label dependencies and diffused semantic features (embeddings for all tasks are presented in Figure 10, Appendix A.6).

Training sets For all three datasets, we sample 10 classes and anonymise them (see Appendix A.6). To create training sets for MultiNERD and GoE-

motion, we generate 3 random support sets consisting of $N = \{3, 4, 5\}$ classes each, and sample $K = \{4, 8, 16\}$ examples for each class from the training split – resulting in a total of 27 $\{N, K\}$ tuples per dataset. For CLINC150, we set $N = 10$ (i.e., all support classes for the seen setting) and sample $K = \{4, 8, 16\}$ for each domain. Since the test split of the dataset already contains OOD examples, we do not need to create an additional OOD test split from non-support classes.

Test sets For each of MultiNERD and GoEmotion, we create 3 test sets by randomly sampling 50 examples from each of the 10 classes in the original test split. As CLINC150 includes OOD domains in its test set, we use these examples for evaluation in the unseen setting (100 examples in total), while for the seen setting we sample 50 examples from each domain of the seen intents.

Evaluation Seen classes are assigned their anonymised label, while unseen ones are labelled as *None of these* (Footnote 1). We train q_{GFN} with $N \cdot K$ episodes and report the mean and standard deviation of F1-scores across the three test sets, separately for seen and unseen classes.

Baselines Following existing baselines in related tasks (Hu et al., 2024; Song et al., 2023; Wang et al., 2024), we compare q_{GFN} to (a) CoT prompting with step-by-step reasoning and a hint for unseen classes (Kojima et al., 2022), and (b) supervised fine-tuning (SFT). CoT helps quantify W-GFN's improvement over reasoning capabilities of pretrained LLMs on unseen examples, while SFT evaluates whether W-GFN surpasses strong in-distribution accuracy. We calculate the 'unseen' performance of the baselines using the same lenient evaluation setting as for W-GFN (Footnote 1).

Construction of episodes Training episodes X contain few-shot demonstrations and a query example. For N support classes with K examples each, we construct $N \cdot K$ episodes, so that every datapoint appears once as a query. At test time, we evaluate only on queries without demonstrations. To provide the anonymised label space required for classification, we take the labels from the training demonstrations (examples in Appendix A.5).

Training details We use Llama-3.2 3B as our base LLM (Grattafiori et al., 2024). We perform extensive hyperparameter sweeps over CoT temperatures (with nucleus sampling), CoT/SFT prompt optimisation, learning and decay rates, epochs, and number of support/query latents (72 latents per training episode) (details in Appendix A.7).

6 Results and Discussion

From Table 1, we observe that W-GFN consistently outperforms all baselines. On MultiNERD, it achieves the highest F1-score in $50/54$ settings for seen classes and $47/54$ for unseen classes. On the more challenging GoEmotion dataset, W-GFN leads in $51/54$ seen settings and $53/54$ unseen settings. While W-GFN marginally underperforms SFT in CLINC150 in the seen setting, SFT suffers substantially from overfitting on the few-shot demonstrations (possibly from surface-level cues provided in test sentences as elaborated in Section 5), failing to generalise in the unseen setting and achieving a score of zero. High F1-scores on MultiNERD are observed with W-GFN with semantically coherent and distinct support class clusters; for instance, combinations such as *Plant, Animal, Mythology, Disease* (in Figure 10a in the appendix) result in higher scores (Table 1a), likely due to tighter latent alignment. In contrast, GoEmotion presents a harder generalisation problem, with comparatively lower scores across all models. This is attributed to both task complexity and less distinct support class clusters (see Figure 10b in the appendix)[7] – an issue that is especially harmful in few-shot setups. Notably, CoT performance on GoEmotion often drops as the number of support classes increases, and adding more few-shot examples provides little improvement, echoing prior results (Wang et al., 2024). SFT consistently performs better than CoT in the seen setting but fails to generalise to unseen classes. Overall, W-GFN is the best-performing model across datasets, maintaining strong results even with as high a number as 10 support classes (CLINC150 dataset).

CoT prompt dependence CoT reasoning is highly prompt-sensitive. Table 3 shows that removing the explicit hint regarding potential unseen examples at test time leads to a clear performance drop, most notably a complete failure of OOD detection on CLINC150 (last row). This suggests that CoT struggles to capture distributional uncertainty without explicit prompt engineering, often tending to follow incorrect reasoning paths. In contrast, W-GFN infers both in- and out-of-distribution

[7] Semantically overlapping classes may result in generated reasoning paths that are more ambiguous (see Appendix A.8).

	K = 4			K = 8			K = 16		
Support Classes	**SFT**	**CoT**	**W-GFN**	**SFT**	**CoT**	**W-GFN**	**SFT**	**CoT**	**W-GFN**
Cosmos, Food, Mythology	$0.37_{0.01}$ $0.01_{0.01}$	$0.37_{0.03}$ $0.54_{0.03}$	$\mathbf{0.58_{0.06}}$ $\mathbf{0.80_{0.01}}$	$0.49_{0.05}$ $0.02_{0.01}$	$0.40_{0.01}$ $0.55_{0.01}$	$\mathbf{0.60_{0.06}}$ $\mathbf{0.67_{0.04}}$	$0.47_{0.01}$ $0.03_{0.02}$	$0.41_{0.04}$ $0.49_{0.01}$	$\mathbf{0.61_{0.05}}$ $\mathbf{0.58_{0.08}}$
Person, Plant, Vehicle	$0.49_{0.01}$ $0.02_{0.01}$	$0.32_{0.03}$ $\mathbf{0.64_{0.03}}$	$\mathbf{0.50_{0.09}}$ $0.58_{0.07}$	$0.53_{0.05}$ $0.06_{0.04}$	$0.34_{0.03}$ $\mathbf{0.58_{0.02}}$	$\mathbf{0.66_{0.07}}$ $0.53_{0.09}$	$0.52_{0.05}$ $0.09_{0.03}$	$0.38_{0.05}$ $\mathbf{0.60_{0.01}}$	$\mathbf{0.63_{0.07}}$ $0.58_{0.10}$
Location, Disease, Time	$0.41_{0.02}$ $0.02_{0.01}$	$0.40_{0.00}$ $0.35_{0.03}$	$\mathbf{0.59_{0.03}}$ $\mathbf{0.70_{0.01}}$	$0.53_{0.0}$ $0.05_{0.03}$	$0.39_{0.02}$ $0.45_{0.03}$	$\mathbf{0.62_{0.07}}$ $\mathbf{0.59_{0.01}}$	$0.48_{0.04}$ $0.01_{0.02}$	$0.39_{0.02}$ $0.39_{0.03}$	$\mathbf{0.64_{0.02}}$ $\mathbf{0.72_{0.04}}$
Disease, Food, Cosmos, Location	$0.48_{0.04}$ $0.08_{0.02}$	$0.47_{0.04}$ $\mathbf{0.47_{0.03}}$	$\mathbf{0.58_{0.05}}$ $0.41_{0.05}$	$0.58_{0.01}$ $0.05_{0.05}$	$0.54_{0.01}$ $0.36_{0.04}$	$\mathbf{0.72_{0.01}}$ $\mathbf{0.55_{0.05}}$	$0.64_{0.06}$ $0.07_{0.06}$	$0.45_{0.04}$ $0.33_{0.02}$	$\mathbf{0.67_{0.03}}$ $\mathbf{0.34_{0.04}}$
Person, Plant, Time, Mythology	$\mathbf{0.55_{0.03}}$ $0.05_{0.05}$	$0.38_{0.02}$ $0.47_{0.03}$	$0.51_{0.05}$ $\mathbf{0.55_{0.08}}$	$0.56_{0.04}$ $0.07_{0.01}$	$0.39_{0.03}$ $0.40_{0.04}$	$\mathbf{0.61_{0.01}}$ $\mathbf{0.60_{0.02}}$	$0.61_{0.03}$ $0.03_{0.02}$	$0.43_{0.04}$ $0.37_{0.01}$	$\mathbf{0.69_{0.02}}$ $\mathbf{0.51_{0.03}}$
Plant, Animal, Mythology Disease	$\mathbf{0.52_{0.04}}$ $0.13_{0.02}$	$0.39_{0.02}$ $0.56_{0.04}$	$0.49_{0.08}$ $\mathbf{0.60_{0.09}}$	$0.54_{0.03}$ $0.01_{0.01}$	$0.40_{0.03}$ $\mathbf{0.65_{0.02}}$	$\mathbf{0.63_{0.04}}$ $\mathbf{0.65_{0.03}}$	$0.56_{0.02}$ $0.09_{0.06}$	$0.35_{0.02}$ $0.60_{0.01}$	$\mathbf{0.67_{0.03}}$ $\mathbf{0.76_{0.04}}$
Animal, Person, Time, Plant, Disease	$\mathbf{0.68_{0.04}}$ $0.01_{0.01}$	$0.46_{0.04}$ $\mathbf{0.52_{0.05}}$	$0.52_{0.04}$ $0.47_{0.05}$	$0.58_{0.04}$ $0.02_{0.02}$	$0.43_{0.03}$ $0.47_{0.04}$	$\mathbf{0.64_{0.03}}$ $\mathbf{0.59_{0.04}}$	$0.67_{0.02}$ $0.02_{0.01}$	$0.41_{0.02}$ $0.44_{0.03}$	$\mathbf{0.74_{0.03}}$ $\mathbf{0.49_{0.06}}$
Plant, Location, Vehicle, Disease, Food	$0.58_{0.03}$ $0.01_{0.01}$	$0.42_{0.04}$ $0.39_{0.01}$	$\mathbf{0.59_{0.05}}$ $\mathbf{0.54_{0.04}}$	$0.59_{0.03}$ $0.05_{0.04}$	$0.41_{0.08}$ $\mathbf{0.52_{0.03}}$	$\mathbf{0.64_{0.04}}$ $0.41_{0.09}$	$0.58_{0.03}$ $0.01_{0.01}$	$0.50_{0.03}$ $0.42_{0.01}$	$\mathbf{0.65_{0.04}}$ $\mathbf{0.45_{0.03}}$
Animal, Mythology, Food, Location, Person	$\mathbf{0.57_{0.03}}$ $0.07_{0.06}$	$0.37_{0.03}$ $0.45_{0.05}$	$0.49_{0.02}$ $\mathbf{0.50_{0.01}}$	$0.60_{0.02}$ $0.10_{0.05}$	$0.39_{0.02}$ $0.41_{0.03}$	$\mathbf{0.71_{0.04}}$ $\mathbf{0.51_{0.06}}$	$0.66_{0.01}$ $0.03_{0.01}$	$0.42_{0.02}$ $0.38_{0.02}$	$\mathbf{0.67_{0.03}}$ $\mathbf{0.64_{0.01}}$

(a) F1-scores for MultiNERD using anonymised labels.

	K = 4			K = 8			K = 16		
Support Classes	**SFT**	**CoT**	**W-GFN**	**SFT**	**CoT**	**W-GFN**	**SFT**	**CoT**	**W-GFN**
Fear, Love, Gratitude	$0.38_{0.01}$ $0.00_{0.00}$	$0.34_{0.01}$ $0.59_{0.03}$	$\mathbf{0.48_{0.08}}$ $\mathbf{0.64_{0.06}}$	$0.41_{0.01}$ $0.00_{0.00}$	$0.31_{0.02}$ $0.40_{0.02}$	$\mathbf{0.52_{0.01}}$ $\mathbf{0.60_{0.08}}$	$0.36_{0.02}$ $0.02_{0.02}$	$0.27_{0.01}$ $0.10_{0.01}$	$\mathbf{0.51_{0.01}}$ $\mathbf{0.59_{0.03}}$
Curiosity, Remorse, Amusement	$0.28_{0.07}$ $0.00_{0.01}$	$0.24_{0.03}$ $0.40_{0.03}$	$\mathbf{0.42_{0.01}}$ $\mathbf{0.52_{0.09}}$	$0.31_{0.02}$ $0.02_{0.01}$	$0.23_{0.02}$ $0.23_{0.02}$	$\mathbf{0.42_{0.04}}$ $\mathbf{0.60_{0.04}}$	$0.37_{0.03}$ $0.02_{0.02}$	$0.23_{0.02}$ $0.10_{0.02}$	$\mathbf{0.45_{0.03}}$ $\mathbf{0.55_{0.04}}$
Gratitude, Anger, Remorse	$0.29_{0.05}$ $0.00_{0.00}$	$0.33_{0.01}$ $0.49_{0.03}$	$\mathbf{0.40_{0.00}}$ $\mathbf{0.51_{0.02}}$	$0.42_{0.02}$ $0.01_{0.01}$	$0.32_{0.02}$ $0.44_{0.01}$	$\mathbf{0.50_{0.06}}$ $\mathbf{0.55_{0.08}}$	$0.35_{0.02}$ $0.00_{0.00}$	$0.32_{0.03}$ $0.29_{0.02}$	$\mathbf{0.54_{0.03}}$ $\mathbf{0.54_{0.05}}$
Admiration, Gratitude, Anger, Fear	$0.30_{0.02}$ $0.00_{0.00}$	$0.27_{0.01}$ $0.31_{0.03}$	$\mathbf{0.40_{0.05}}$ $\mathbf{0.55_{0.08}}$	$0.45_{0.03}$ $0.00_{0.00}$	$0.34_{0.02}$ $0.29_{0.03}$	$\mathbf{0.56_{0.02}}$ $\mathbf{0.50_{0.04}}$	$0.38_{0.05}$ $0.00_{0.00}$	$0.36_{0.02}$ $0.21_{0.09}$	$\mathbf{0.54_{0.04}}$ $\mathbf{0.35_{0.07}}$
Fear, Remorse, Love, Admiration	$0.31_{0.03}$ $0.00_{0.00}$	$0.28_{0.01}$ $\mathbf{0.46_{0.06}}$	$\mathbf{0.54_{0.02}}$ $0.37_{0.08}$	$0.36_{0.03}$ $0.00_{0.00}$	$0.31_{0.01}$ $0.24_{0.01}$	$\mathbf{0.52_{0.04}}$ $\mathbf{0.54_{0.09}}$	$0.40_{0.02}$ $0.00_{0.00}$	$0.23_{0.01}$ $0.27_{0.04}$	$\mathbf{0.50_{0.07}}$ $\mathbf{0.50_{0.03}}$
Love, Amusement, Curiosity, Sadness	$0.31_{0.02}$ $0.00_{0.00}$	$0.29_{0.05}$ $0.38_{0.04}$	$\mathbf{0.48_{0.07}}$ $\mathbf{0.41_{0.07}}$	$0.41_{0.05}$ $0.00_{0.00}$	$0.24_{0.02}$ $0.23_{0.01}$	$\mathbf{0.42_{0.02}}$ $\mathbf{0.41_{0.08}}$	$0.38_{0.06}$ $0.03_{0.02}$	$0.28_{0.02}$ $0.11_{0.03}$	$\mathbf{0.47_{0.01}}$ $\mathbf{0.43_{0.01}}$
Fear, Remorse, Admiration, Curiosity, Anger	$0.33_{0.02}$ $0.00_{0.00}$	$0.32_{0.03}$ $0.35_{0.03}$	$\mathbf{0.42_{0.01}}$ $\mathbf{0.39_{0.03}}$	$0.38_{0.01}$ $0.00_{0.01}$	$0.28_{0.00}$ $0.13_{0.03}$	$\mathbf{0.43_{0.01}}$ $\mathbf{0.50_{0.06}}$	$\mathbf{0.44_{0.08}}$ $0.00_{0.00}$	$0.27_{0.02}$ $0.11_{0.02}$	$\mathbf{0.44_{0.05}}$ $\mathbf{0.40_{0.03}}$
Love, Anger, Gratitude, Curiosity, Amusement	$0.45_{0.04}$ $0.01_{0.01}$	$0.29_{0.02}$ $0.38_{0.03}$	$\mathbf{0.52_{0.03}}$ $\mathbf{0.62_{0.01}}$	$\mathbf{0.49_{0.01}}$ $0.01_{0.01}$	$0.29_{0.01}$ $0.17_{0.02}$	$0.48_{0.04}$ $\mathbf{0.45_{0.07}}$	$\mathbf{0.54_{0.01}}$ $0.01_{0.01}$	$0.30_{0.01}$ $0.16_{0.03}$	$0.50_{0.02}$ $\mathbf{0.36_{0.12}}$
Remorse, Love, Sadness, Admiration, Gratitude	$0.32_{0.02}$ $0.00_{0.00}$	$0.23_{0.02}$ $0.31_{0.01}$	$\mathbf{0.37_{0.03}}$ $\mathbf{0.52_{0.06}}$	$0.35_{0.02}$ $0.01_{0.01}$	$0.29_{0.03}$ $0.21_{0.05}$	$\mathbf{0.53_{0.05}}$ $\mathbf{0.63_{0.06}}$	$0.42_{0.04}$ $0.00_{0.00}$	$0.28_{0.02}$ $0.20_{0.03}$	$\mathbf{0.43_{0.01}}$ $\mathbf{0.51_{0.06}}$

(b) F1-scores for GoEmotion using anonymised labels.

	K = 4			K = 8			K = 16		
Support Classes	**SFT**	**CoT**	**W-GFN**	**SFT**	**CoT**	**W-GFN**	**SFT**	**CoT**	**W-GFN**
Banking, Credit Cards, Dining, Home, Auto/Commute, Travel, Utility, Work, Small Talk, Meta	$\mathbf{0.54_{0.03}}$ $0.00_{0.00}$	$0.41_{0.01}$ $0.25_{0.02}$	$0.50_{0.03}$ $\mathbf{0.26_{0.02}}$	$\mathbf{0.63_{0.02}}$ $0.00_{0.00}$	$0.47_{0.02}$ $0.21_{0.03}$	$0.55_{0.02}$ $\mathbf{0.30_{0.01}}$	$\mathbf{0.64_{0.02}}$ $0.00_{0.00}$	$0.52_{0.01}$ $0.19_{0.02}$	$0.60_{0.02}$ $\mathbf{0.26_{0.02}}$

(c) F1-scores for CLINC150 using anonymised labels.

Table 1: Each table cell contains two rows: the first row shows the average F1-score for seen classes; the second row shows the average F1-score for unseen classes. Entries in green indicate the highest score.

likelihoods through learned latent-to-label transitions, quantitatively demonstrating its ability to model posterior uncertainty in a principled and data-driven manner (see Appendix A.8 for complete set of results).

Comparison with Llama-3.3 70B CoT prompting is an emergent capability that typically becomes robust in very large LLMs (Kojima et al., 2022; Wei et al., 2022a,b). We therefore compare W-GFN (using Llama-3.2 3B) to CoT reasoning with the substantially larger Llama-3.3 70B.[8] Table 2 shows that W-GFN (3B) often approaches the performance of Llama-3.3 70B for high-complexity tasks, reaching $\geq 80\%$ of its performance on GoEmotion. Llama 3.3 70B generalises well on tasks such as MultiNERD (where structured, pattern-driven classification is effective) and CLINC150

[8] A model $\sim$ 23 times larger than Llama-3.2 3B, trained on substantially more data, and tuned for complex reasoning tasks, demonstrating 150% pre-training improvement over Llama-3.2 3B on GPQA-Diamond (Rein et al., 2024).

Support Classes	CoT (3B)	W-GFN (3B)	CoT (70B)	%
Fear, Love, Gratitude	$0.31_{0.02}$ $0.40_{0.02}$	$0.52_{0.01}$ $0.60_{0.08}$	$0.63_{0.02}$ $0.76_{0.02}$	81%
Admiration, Gratitude, Anger, Fear	$0.34_{0.02}$ $0.29_{0.03}$	$0.56_{0.02}$ $0.50_{0.04}$	$0.57_{0.03}$ $0.51_{0.02}$	98%
Remorse, Love, Sadness, Admiration, Gratitude	$0.29_{0.03}$ $0.21_{0.05}$	$0.53_{0.05}$ $0.63_{0.06}$	$0.66_{0.03}$ $0.76_{0.02}$	84%
Disease, Food, Cosmos, Location	$0.54_{0.01}$ $0.36_{0.04}$	$0.72_{0.01}$ $0.55_{0.05}$	$0.90_{0.02}$ $0.93_{0.03}$	70%
Banking, Credit Cards, Dining, Home, Auto/Commute, Travel, Utility, Work, Small Talk, Meta	$0.47_{0.02}$ $0.21_{0.03}$	$0.55_{0.02}$ $0.30_{0.01}$	$0.80_{0.01}$ $0.45_{0.05}$	68%

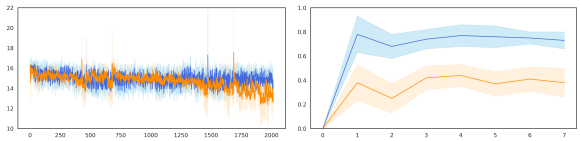
Table 2: % shows W-GFN performance (Llama-3.2 3B) relative to CoT (Llama-3.3 70B) (8 examples per class).

Support Classes	W-GFN (3B)	CoT$^-$ (3B)	CoT (3B)
Admiration, Gratitude, Anger, Fear	$0.56_{0.02}$ $0.50_{0.04}$	$0.35_{0.01}$ $0.11_{0.02}$	$0.34_{0.02}$ $0.29_{0.03}$
Disease, Food, Cosmos, Location	$0.72_{0.01}$ $0.55_{0.05}$	$0.51_{0.03}$ $0.09_{0.02}$	$0.54_{0.01}$ $0.36_{0.04}$
Banking, Credit Cards, Dining, Home, Auto/Commute, Travel, Utility, Work, Small Talk, Meta	$0.55_{0.02}$ $0.30_{0.01}$	$0.49_{0.02}$ $0.00_{0.00}$	$0.47_{0.02}$ $0.21_{0.03}$

Table 3: CoT without (COT$^-$) and with (COT) an out-of-distribution hint, using 8 examples per class.

(where surface-level cues in both few-shot demonstrations and query examples aid open-set classification); however, it performs less robustly on more complex tasks such as GoEmotion. Remarkably, W-GFN can sometimes outperform Llama-3.3 70B CoT reasoning when the OOD hint is omitted from the prompt (see Appendix A.8). This highlights the efficiency and generalisation capability of W-GFN, positioning it as a practical alternative to extremely large models. Inference with full-precision Llama-3.3 70B is highly resource-intensive, requiring 4 NVIDIA A100 40GB GPUs (using a total of 150GB GPU memory) and ≥ 1 min per example for CoT reasoning, whereas a trained W-GFN runs on a single NVIDIA A100 40GB GPU using only 7GB GPU memory, with inference times as low as $5-8$ sec/example.

Impact of Wasserstein offset We study the effect of training with and without the Wasserstein-style offset (i.e., constant reward), using the support classes *Disease, Food, Cosmos, Location*, and training q_{GFN} for 2 epochs. Figure 5 shows that incorporating the offset stabilises training and improves validation performance. It also suggests that constant reward shaping can lead to overfitting: the model without the offset reaches a validation accuracy of a max of only $0.45_{0.10}$, compared to $0.77_{0.05}$ with the Wasserstein offset. The smooth Wasserstein interpolation helps q_{GFN} assign similar rewards to semantically related latent sequences without abrupt jumps in the reward signal, which



(a) $\log \mathcal{L}_{SubTB}(Z;\theta)$ (b) Validation accuracies

Figure 5: Training loss and validation accuracy without (orange) and with (blue) the Wasserstein offset.

	8	16	32	48	64	128
8	0.69	0.66	0.69	0.7	0.67	0.65
16	0.7	0.71	0.68	0.68	0.65	0.69
32	0.72	0.66	0.72	0.64	0.64	0.63
48	0.71	0.69	0.66	0.69	0.66	0.66
64	0.72	0.71	0.69	0.68	0.66	0.64
128	0.7	0.71	0.7	0.68	0.66	0.69

Figure 6: Mean validation accuracy over seen/unseen classes with varying support (↓) and query latents (→).

encourages learning a smooth and generalisable distribution over latent sequences.

Latent allocation tradeoff Figure 6 presents an ablation study varying the number of support and query latents (latents describing few-shot demonstrations and the query example respectively). We use the support classes *Disease, Food, Cosmos, Location* and train q_{GFN} for two epochs. We find that strong generalisation is primarily driven by support latents, which capture class-level abstractions and are particularly effective at mitigating noise. Query latents provide complementary, instance-specific reasoning that further boost performance. The results suggest that (class-level) support latents combined with a small number of (instance-specific) query latents yield the best tradeoff.

7 Conclusions

We demonstrate that Wasserstein-GFN (W-GFN) enables robust few-shot classification and principled out-of-distribution detection within a unified framework, even with small LLMs. Our approach substantially improves reasoning performance and provides a viable alternative to much larger models. We also release our code to support further research in this area.

Limitations

Computationally heavy reward signal Training q_{GFN} is computationally demanding and scales with the length of the latent sequence

Z. Specifically, computing the log reward difference $R(i,j) = \Sigma_t \log p_{LLM}(XZ_{1:i}Y) - \Sigma_t \log p_{LLM}(XZ_{1:j}Y)$ across all token pairs $\{i,j\}$ in the complete latent sequence (from Equation 1) becomes intractable for very long sequences. To address this, we cap the maximum length of Z to 15 tokens. While this works well in our domains, this restriction could reduce representational capacity for other complex domains such as commonsense reasoning, mathematics, or coding. A potential remedy is to approximate the reward landscape with Gaussian Processes (see Appendix A.8).

References

Lazar Atanackovic, Alexander Tong, BO WANG, Leo J Lee, Yoshua Bengio, and Jason Hartford. 2023. DynGFN: Towards bayesian inference of gene regulatory networks with GFlownets. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2020a. Learning to few-shot learn across diverse natural language classification tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5108–5123, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020b. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534, Online. Association for Computational Linguistics.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J. Hu, Mo Tiwari, and Emmanuel Bengio. 2023. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55.

Arth Bohra, Govert Verkes, Artem Harutyunyan, Pascal Weinberger, and Giovanni Campagna. 2023. BYOC: Personalized few-shot classification with co-authored class descriptions. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13999–14015, Singapore. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *International Conference on Learning Representations*.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Junfan Chen, Richong Zhang, Junchi Chen, and Chunming Hu. 2024. Open-set semi-supervised text classification via adversarial disagreement maximization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2170–2180, Bangkok, Thailand. Association for Computational Linguistics.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *Preprint*, arXiv:2307.08691.

Tristan Deleu, Mizu Nishikawa-Toomey, Jithendaraa Subramanian, Nikolay Malkin, Laurent Charlin, and Yoshua Bengio. 2023. Joint bayesian inference of graphical structure and parameters with a single generative flow network. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. GoEmotions: A dataset of fine-grained emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4040–4054, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Chuanxing Geng, Sheng-Jun Huang, and Songcan Chen. 2021. Recent advances in open set recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(10):3614–3631.

Marc G. Genton. 2002. Classes of kernels for machine learning: a statistics perspective. *J. Mach. Learn. Res.*, 2:299–312.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Kang He, Yinghan Long, and Kaushik Roy. 2024. Prompt-based bias calibration for better zero/few-shot learning of language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12673–12691, Miami, Florida, USA. Association for Computational Linguistics.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. 2022. Meta-Learning in Neural Networks: A Survey . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 44(09):5149–5169.

Edward J Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, Guillaume Lajoie, Yoshua Bengio, and Nikolay Malkin. 2024. Amortizing intractable inference in large language models. In *The Twelfth International Conference on Learning Representations*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24(1).

Sashank J. Reddi, Sobhan Miryoosefi, Stefani Karp, Shankar Krishnan, Satyen Kale, Seungyeon Kim, and Sanjiv Kumar. 2023. Efficient training of language models using few-shot learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 14553–14568. PMLR.

Ke Ji, Yixin Lian, Jingsheng Gao, and Baoyuan Wang. 2023. Hierarchical verbalizer for few-shot hierarchical text classification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2918–2933, Toronto, Canada. Association for Computational Linguistics.

Masahiro Kaneko, Graham Neubig, and Naoaki Okazaki. 2024. Solving NLP problems through human-system collaboration: A discussion-based approach. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1644–1658, St. Julian's, Malta. Association for Computational Linguistics.

Minsu Kim, Sanghyeok Choi, Jiwoo Son, Hyeonah Kim, Jinkyoo Park, and Yoshua Bengio. 2024. Ant colony sampling with gflownets for combinatorial optimization. *CoRR*, abs/2403.07041.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. *Preprint*, arXiv:1312.6114.

Atsushi Kojima. 2024. Sub-table rescorer for table question answering. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15422–15427, Torino, Italia. ELRA and ICCL.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Salem Lahlou, Tristan Deleu, Pablo Lemos, Dinghuai Zhang, Alexandra Volokhova, Alex Hernández-García, Lena Nehale Ezzine, Yoshua Bengio, and Nikolay Malkin. 2023. A theory of continuous generative flow networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 18269–18300. PMLR.

Andrew Lampinen, Ishita Dasgupta, Stephanie Chan, Kory Mathewson, Mh Tessler, Antonia Creswell, James McClelland, Jane Wang, and Felix Hill. 2022. Can language models learn from explanations in context? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 537–563, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Bo Liu, Hao Kang, Haoxiang Li, Gang Hua, and Nuno Vasconcelos. 2020. Few-shot open-set recognition using meta-learning. *Preprint*, arXiv:2005.13713.

Bo Liu, Li-Ming Zhan, Zexin Lu, Yujie Feng, Lei Xue, and Xiao-Ming Wu. 2024a. How good are LLMs at out-of-distribution detection? In *Proceedings of*

the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pages 8211–8222, Torino, Italia. ELRA and ICCL.

Xinyue Liu, Yunlong Gao, Linlin Zong, and Bo Xu. 2024b. Improve meta-learning for few-shot text classification with all you can acquire from the tasks. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 223–235, Miami, Florida, USA. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. 2023. Learning GFlownets from partial episodes for improved convergence and stability.

Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. 2022. Trajectory balance: Improved credit assignment in GFlownets. In *Advances in Neural Information Processing Systems*.

A. Podol'skii, Dmitry Lipin, Andrey Bout, Ekaterina Artemova, and Irina Piontkovskaya. 2021. Revisiting mahalanobis distance for transformer-based out-of-domain detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:13675–13682.

Carl Edward Rasmussen. 2004. *Gaussian Processes in Machine Learning*, pages 63–71. Springer Berlin Heidelberg, Berlin, Heidelberg.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Julien Roy, Pierre-Luc Bacon, Christopher Pal, and Emmanuel Bengio. 2023. Goal-conditioned gflownets for controllable multi-objective molecular design. *Preprint*, arXiv:2306.04620.

Avyav Singh, Ekaterina Shutova, and Helen Yannakoudakis. 2024. Learning new tasks from a few examples with soft-label prototypes. In *Proceedings of the 9th Workshop on Representation Learning for NLP (RepL4NLP-2024)*, pages 215–236, Bangkok, Thailand. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. *Preprint*, arXiv:1703.05175.

Xiaoshuai Song, Keqing He, Pei Wang, Guanting Dong, Yutao Mou, Jingang Wang, Yunsen Xian, Xunliang Cai, and Weiran Xu. 2023. Large language models meet open-world intent discovery and recognition: An evaluation of ChatGPT. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10291–10304, Singapore. Association for Computational Linguistics.

Simone Tedeschi and Roberto Navigli. 2022. MultiNERD: A multilingual, multi-genre and fine-grained dataset for named entity recognition (and disambiguation). In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 801–812, Seattle, United States. Association for Computational Linguistics.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6309–6318, Red Hook, NY, USA. Curran Associates Inc.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Vijay Viswanathan, Kiril Gashteovski, Kiril Gashteovski, Carolin Lawrence, Tongshuang Wu, and Graham Neubig. 2024. Large language models enable few-shot clustering. *Transactions of the Association for Computational Linguistics*, 12:321–333.

Pei Wang, Keqing He, Yejie Wang, Xiaoshuai Song, Yutao Mou, Jingang Wang, Yunsen Xian, Xunliang Cai, and Weiran Xu. 2024. Beyond the known: Investigating LLMs performance on out-of-domain intent detection. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2354–2364, Torino, Italia. ELRA and ICCL.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le

Scao, Sylvain Gugger, and 3 others. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Hong Xu, Keqing He, Yuanmeng Yan, Sihong Liu, Zijun Liu, and Weiran Xu. 2020. A deep generative distance-based classifier for out-of-domain detection with mahalanobis space. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1452–1460, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. 2024. Generalized out-of-distribution detection: A survey. *International Journal of Computer Vision*, 132(12):5635–5662.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2023. Diffusion models: A comprehensive survey of methods and applications. *ACM Comput. Surv.*, 56(4).

Yue Yu, Rongzhi Zhang, Ran Xu, Jieyu Zhang, Jiaming Shen, and Chao Zhang. 2023. Cold-start data selection for better few-shot language model fine-tuning: A prompt-based uncertainty propagation approach. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2499–2521, Toronto, Canada. Association for Computational Linguistics.

A. Zhang, T. Z. Xiao, W. Liu, R. Bamler, and D. Wischik. 2025. Your finetuned large language model is already a powerful out-of-distribution detector. In *The 28th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Dinghuai Zhang, Hanjun Dai, Nikolay Malkin, Aaron Courville, Yoshua Bengio, and Ling Pan. 2023. Let the flows tell: Solving graph combinatorial problems with GFlownets. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Wenxuan Zhou, Fangyu Liu, and Muhao Chen. 2021. Contrastive out-of-distribution detection for pre-trained transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1100–1111, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yiheng Zhu, Jialu Wu, Chaowen Hu, Jiahuan Yan, Chang-Yu Hsieh, Tingjun Hou, and Jian Wu. 2023. Sample-efficient multi-objective molecular optimization with GFlownets. In *Thirty-seventh Conference on Neural Information Processing Systems*.

A Appendix

A.1 Prompts for Sparse-Shot Learning

An example prompt is provided in Figure 7. Here, we provide 3 classes with 3 few-shot demonstrations each and the objective is to classify the query example correctly within the seen classes.

A.2 Generative Flow Networks

Generative Flow Networks (GFNs) are a class of models designed to sample from complex distributions over structured objects by modeling a stochastic process over sequences (trajectories) of actions. A key objective in training GFNs is to ensure that the distribution over complete sequences (terminal states) induced by the policy is proportional to a given *reward function* $R(x)$.

Key Condition: Flow Matching and Detailed Balance GFNs define a flow $F(s, a)$ over state-action pairs, where the incoming and outgoing flows must match under the condition:

$$\sum_{a' \in \text{pred}(s)} F(s', a') = \sum_{a \in \text{succ}(s)} F(s, a)$$

A stronger condition that ensures global consistency is the *detailed balance condition*, which requires:

$$F(s, a) = F(s', a')$$

for any forward and backward pair $s \to s'$. As discussed before, this implies a form of symmetry and equilibrium between forward and backward flows, leading to a stationary distribution which can be sampled from. Several loss functions have been proposed to train GFNs to approximate the desired reward-distribution proportionality.

Trajectory Balance (TB) This loss ensures that the total flow along a trajectory matches the reward at the terminal state. For a trajectory $\tau = (s_0, a_0, \ldots, s_T)$ that ends in terminal state x, the condition is:

$$\log R(x) = \log Z + \sum_{t=0}^{T-1} \log P_F(a_t|s_t) - \sum_{t=1}^{T} \log P_B(a_{t-1}|s_t)$$

Here, P_F and P_B are the forward and backward policies, and Z is a learned normalisation constant.

In the sentence 'He generally skated with captain Jonathan Toews on the team 's first line.' the entity ['Jonathan Toews'] represents Class A.
In the sentence 'He is buried in Vilnius Cathedral together with his brother Karigaila, who died in the civil war in 1390.' the entity ['Karigaila'] represents Class A.
In the sentence 'Minister Margaret Thatcher complained when, by adroit image editing, the programme implied she had crashed a car.' the entity ['Margaret Thatcher'] represents Class A.
In the sentence 'In the job interview he encounters eccentric owner (Gunnar Björnstrand), who informs him the duties of job and possible promotion to a projectionist in the future.' the entity ['Gunnar Björnstrand'] represents Class A.
In the sentence 'The culture can be naturally captured from the wild, by mixing rice flour with ground spices (include garlic, pepper, chili, cinnamon), cane sugar or coconut water, slices of ginger or ginger extract, and water to make a dough.' the entities ['pepper', 'chili', 'cinnamon', 'ginger'] represent Class M.
In the sentence '" Pouteria virescens " is a species of plant in the family Sapotaceae.' the entities ['plant', 'Sapotaceae'] represent Class M.
In the sentence 'When possible, three washings are performed : first with water infused with plum leaves, then with water infused with camphor, and lastly with purified water.' the entity ['plum'] represents Class M.
In the sentence 'The amount can be less than one percent in animals consuming less digestible plants, and it can be as high as forty percent in zooplankton consuming phytoplankton.' the entity ['phytoplankton'] represents Class M.
In the sentence 'The class was gradually repainted from 1983 as the V / Line logo and colour scheme was introduced.' the entity ['V / Line'] represents Class O.
In the sentence 'This record was finally broken in 2004 by a Porsche Carrera GT, which did it in.' the entity ['Porsche Carrera GT'] represents Class O.
In the sentence 'The engine made its first flight aboard a Gotha Go 145 on 30 April 1941.' the entity ['Gotha Go 145'] represents Class O.
In the sentence 'He was the backup of Klaus-Dietrich Flade for the Soyuz TM-14 mission.' the entity ['Soyuz TM-14'] represents Class O. Which class does the entity ['James Wyatt'] in the sentence 'An extension to the north designed by James Wyatt was added in 1785.' represent? Think step by step. Reply with 'None of these' if the answer does not fall in any class.

Figure 7: Example input for the NER task. We provide three examples per class (total of three classes) here.

The TB loss (Malkin et al., 2022) is defined as

$$\mathcal{L}_{\text{TB}} = \mathbb{E}_{\tau\sim P_F}\left[(\log R(x) - \log Z - \sum_{t=0}^{T-1}\log P_F(a_t|s_t) + \sum_{t=1}^{T}\log P_B(a_{t-1}|s_t)\right)^2\Bigg]$$

Subtrajectory Balance This loss generalises TB by applying balance conditions to subparts of a trajectory. For subtrajectories $(s_i, \ldots, s_j)$:

$$\mathcal{L}(Z) = \sum_{0\le i<j\le T}\left(\log\frac{R(s_i)}{R(s_j)} + \sum_{t=i+1}^{j}\log P_F(a_t|s_t) - \sum_{t=i}^{j-1}\log P_B(a_t|s_{t+1})\right)^2$$

This provides richer supervision and enables off-policy training from partial data. Note that existing work (Hu et al., 2024) uses a variation of this loss provided previously in Equation 1 for language models as

$$\mathcal{L}_{SubTB}(Z;\theta) = \sum_{1<j\le n}\sum_{0\le i<j}\left[\log\frac{R(z_{1:i}\top)\prod_{k=i+1}^{j}q(z_k|z_{1:k-1})q(\top|z_{1:j})}{R(z_{1:j}\top)q(\top|z_{1:i})}\right]^2$$

Flow Matching Loss Another approach is directly minimising the squared difference between incoming and outgoing flows at each state (Bengio et al., 2023) which is defined as

$$\mathcal{L}_{\text{FM}} = \sum_{s\in\mathcal{S}}\left[\sum_{a'\in \text{p}(s)}F(s',a') - \sum_{a\in \text{s}(s)}F(s,a)\right]^2$$

where $p(s_t)$ and $s(s_t)$ refer to predecessor and successor states of s_t. This enforces the local flow-matching condition but may be harder to train due to lack of global context.

Detailed Balance Loss To directly enforce the detailed balance constraint between forward and backward transitions, there is another loss function defined which satisfies all conditions of a GFN which is called the *detailed balance* loss. the detailed balance loss is defined as

$$\mathcal{L}_{\text{DB}} = \sum_{(s,a,s')}\left(\log F(s,a) - \log F_B(s',a')\right)^2$$

which shows success in Bayesian structure learning (Lahlou et al., 2023) amongst other tasks.

Class A broadly represents names of people.
Class A broadly represents famous people, including politicians and soldiers.
Class A broadly represents well-known people in history.
.
.

(a) Sampling support conditional latents

The entity ['James Wyatt'] in the sentence 'An extension to the north designed by James Wyatt was added in 1785.' broadly represents an English architect famous for his neo-Gothic work.
The entity ['James Wyatt'] in the sentence 'An extension to the north designed by James Wyatt was added in 1785.' broadly represents a British architect who designed many buildings.
The entity ['James Wyatt'] in the sentence 'An extension to the north designed by James Wyatt was added in 1785.' broadly represents James Wyatt, the English architect active in the eighteenth century.
.
.

(b) Sampling query conditional latents

The test entity broadly represents names of people.
The test entity broadly represents famous people, including politicians and soldiers.
The test entity broadly represents well-known people in history.
The test entity broadly represents an English architect famous for his neo-Gothic work.
The test entity broadly represents a British architect who designed many buildings.
The test entity broadly represents James Wyatt, the English architect active in the eighteenth century.
.
.

(c) Final conditional prior latent sequences used to train q_{GFN}

Figure 8: Support and query latents for the example in Figure 7

A.3 Sampling support latents and query latents

For the prompt in Figure 7, the sampled support latents and query labels can be seen in Figure 8. For training the GFN, we use only the descriptive part of the latent sequence and replace the prefix with "The test entity" as demonstrated in Figure 8c. At test time, we only sample query latents as we do not have access to the correct support set.

A.4 Pitfalls of a positive offset in the reward

Given a sub-trajectory loss function for a GFN as

$$\mathcal{L}_{SubTB}(Z;\theta) = \sum_{1<j\leq n} \sum_{0\leq i<j} \left[\log \frac{R(z_{1:i}\top)\prod_{k=i+1}^{j} q(z_k|z_{1:k-1})q(\top|z_{1:j})}{R(z_{1:j}\top)q(\top|z_{1:i})}\right]^2$$

where $q(\cdot|\cdot)$ refers to $q_{GFN}(\cdot|\cdot)$. Let $\pi_{i\to j} := \prod_{k=i+1}^{j} q_{GFN}(z_k|z_{1:k-1})$. The loss can thus be rewritten as:

$$\mathcal{L}(Z;\theta) = \sum_{0\leq i<j\leq n} \left(\log\left(\frac{R(z_{1:i}\top)}{R(z_{1:j}\top)}\cdot \frac{\pi_{i\to j}\cdot q_{GFN}(\top|z_{1:j})}{q_{GFN}(\top|z_{1:i})}\right)\right)^2$$

Consider a case where $\log R(Z_{1:i}\top)$ is higher for the correct label than an incorrect label at token i (i.e.: $\Sigma \log p_{LLM}(XZ_{1:i}Y_{correct}) > \Sigma p_{LLM}\log(XZ_{1:i}Y_{incorrect})$) but it is lower at token j (i.e.: $\Sigma p_{LLM}\log(XZ_{1:j}Y_{correct}) < \Sigma p_{LLM}\log(XZ_{1:j}Y_{incorrect})$) which is a scenario we came across while training q_{GFN}. Thus, we would need to add a positive scalar offset $\Delta > 0$ to the log-probabilities of the reward to the correct trajectory $XZ_{1:j}Y_{correct}$ to get :

$$\mathcal{L}(Z;\theta) = \sum_{0\leq i<j\leq n} \left(\log \frac{R(z_{1:i}\top)}{R(z_{1:j}\top)+\delta}\cdot \frac{\pi_{i\to j}\cdot q_{GFN}(\top|z_{1:j})}{q_{GFN}(\top|z_{1:i})}\right)^2$$

Parameter	Search Space	SFT	CoT	W-GFN
Sampling temperature	$\{0.6, 0.7, 0.8, 0.9\}$	$\{0.7, 0.8\}^*$	$\{0.7, 0.8, 0.9\}^*$	0.7
LORA rank	$\{32, 64, 256\}$	32	–	256
LORA α	$\{4, 8, 16\}$	4	–	16
Learning rate	$\{1e-6 : 1e-4\}$	$\{1e-5 : 1e-4\}^*$	–	$\{1e-6 : 1e-4\}^*$
Learning rate scheduler	$\{cosine, uniform\}$	$uniform$	–	$cosine$
Dropout	$\{0.0 : 0.5\}$	0.0	–	0.0
Epochs	$\{1, 3, 5, 10\}$	$\{3 : 5\}$	–	10
Batch size for latents	$\{16, 24, 32\}$	–	–	$\{16, 24, 32\}^*$
Length of latent Z	$\{10, 15\}$	–	–	15
Support latents	$\{8, 24, 36, 48\}$	–	–	$\{36, 48\}$
Query latents	$\{8, 24, 36, 48\}$	–	–	$\{24, 36\}$
Test latents	$\{9\}$	–	–	$\{9\}$
$k(x, x')$	RBF, Matérn$\{3/2, 5/2\}$	–	–	Matérn $\{5/2\}$
Δ	$\{50, 80, 100\}$	–	–	$\{50, 80\}^*$

Table 4: Hyperparameters for training all methods

for a constant δ such that $\log(R(z_{1:j}\top) + \delta) = \log R(z_{1:j}\top) + \Delta$. Note that $R(z_{1:j}\top) < R(z_{1:i}\top)$ therefore $\log R(z_{1:j}\top) < \log R(z_{1:i}\top) < 0$. From the definition of a GFN (Bengio et al., 2023), $\mathcal{L}(Z;\theta)$ minimises when

$$\frac{R(z_{1:i}\top)}{R(z_{1:j}\top) + \delta} \cdot \frac{\pi_{i\to j} \cdot q_{GFN}(\top|z_{1:j})}{q_{GFN}(\top|z_{1:i})} \to 1$$

Suppose that after adding δ, we have $R(z_{1:j}\top) + \delta > R(z_{1:i}\top)$ (a quite realistic scenario as $R(z_{1:i}\top) = \Pi p_{LLM}(XZ_{1:i}Y) << 1$), therefore at convergence we should have

$$\frac{\pi_{i\to j} \cdot q_{GFN}(\top|z_{1:j})}{q_{GFN}(\top|z_{1:i})} > 1$$

However, this condition can only be satisfied when $q_{GFN}(\top|z_{1:i}) \to 0$ because $\pi_{i\to j} \cdot q_{GFN}(\top|z_{1:j}) = \prod_{k=i+1}^{j} q_{GFN}(z_k|z_{1:k-1}) \cdot q_{GFN}(\top|z_{1:j}) \to 0$ especially for long token sequences between i and j. This, in turn, causes instability in $\nabla_\theta \mathcal{L}(Z;\theta)$ due to a very high negative value of $\log q_{GFN}(\top|z_{1:i})$ which in turn causes an underflow error. Note that a backward policy function $\pi^B_{j\to i}$ can stabilise training as a general GFN typically has forward and backward policy functions, however, we have set the backward policy function $\pi^B(s_t|s_{t+1}) = 1$ for q_{GFN} which approximates language models.

A.5 Training episode format for q_{GFN}

Similar to the prompt format for CoT reasoning in Figure 7, we include few-shot demonstrations and a query training example. We append all latents generated and for each latent, we append the complete label set. The GFN objective is thus to train with the correct label sequence $XZY_{correct}$ with the offsets added to the reward and other sequences $XZY_{incorrect}$ are trained with their reward signals without any offset attached – thus we can recover high-probability logits using the logit normalisation trick in Section 4. We provide an example in Figure 9b.

A.6 Dataset details

Training dataset We use three tasks to evaluate the performance of q_{GFN} on open-set classification – namely, Named Entity Recognition (NER) using the dataset MultiNERD (Tedeschi and Navigli, 2022), fine-grained emotion classification using the dataset GoEmotion (Demszky et al., 2020) as well as fine-grained intent classification on the dataset CLINC150 (Larson et al., 2019). We sample ten random classes from MultiNERD (*Person, Location, Animal, Cosmos, Disease, Food, Mythology, Plant, Vehicle, Time*) which form the complete test distribution as well as GoEmotion (*Neutral, Amusement, Admiration, Fear, Gratitude, Love, Sadness, Anger, Curiosity, Remorse*) and CLINC150 (*Banking, Credit Cards, Kitchen/Dining, Home, Auto/-Commute, Travel, Utility, Work, Small Talk, Meta*). We then anonymise all classes by using a single capital letter and train a different q_{GFN} per set of support classes with a varying number of examples per class (we define $K \in \{4, 8, 16\}$).

Validation dataset We use the same validation dataset which consists of 10 uniformly sampled points per class (therefore, 100 data points in total) for both MultiNERD, GoEmotion and CLINC150. We use this validation set for W-GFN and our other

In the sentence 'He is buried in Vilnius Cathedral together with his brother Karigaila, who died in the civil war in 1390.' the entity ['Karigaila'] represents Class A.
In the sentence 'In the job interview he encounters eccentric owner (Gunnar Björnstrand), who informs him the duties of job and possible promotion to a projectionist in the future.' the entity ['Gunnar Björnstrand'] represents Class A.
In the sentence 'The culture can be naturally captured from the wild, by mixing rice flour with ground spices (include garlic, pepper, chili, cinnamon), cane sugar or coconut water, slices of ginger or ginger extract, and water to make a dough.' the entities ['pepper', 'chili', 'cinnamon', 'ginger'] represent Class M.
In the sentence '" Pouteria virescens " is a species of plant in the family Sapotaceae.' the entities ['plant', 'Sapotaceae'] represent Class M.
In the sentence 'The engine made its first flight aboard a Gotha Go 145 on 30 April 1941.' the entity ['Gotha Go 145'] represents Class O.
In the sentence 'He was the backup of Klaus-Dietrich Flade for the Soyuz TM-14 mission.' the entity ['Soyuz TM-14'] represents Class O. Which class does the entity ['James Wyatt'] in the sentence 'An extension to the north designed by James Wyatt was added in 1785.' represent?

(a) We provide two examples per class (total of three classes) here and we define this as the input X.

The test entity broadly represents names of people, thus, it belongs to Class A. +
The test entity broadly represents names of people, thus, it belongs to Class M. ∘
The test entity broadly represents names of people, thus, it belongs to Class O. ∘
The test entity broadly represents famous people, including politicians and soldiers, thus, it belongs to Class A. +
The test entity broadly represents famous people, including politicians and soldiers, thus, it belongs to Class M. ∘
The test entity broadly represents famous people, including politicians and soldiers, thus, it belongs to Class O. ∘
...

(b) Gathered latents (see Figure 8c) attached to the list of labels to form $ZY|X$. Sequences marked with + are trained with the reward offset while sequences marked with ∘ are trained with an unchanged reward signal.

Figure 9: Training episode format for q_{GFN}

baselines (CoT and SFT).

Task complexity We sample few-shot support sets ranging from 3 to 5 classes (inclusive) for MultiNERD and GoEmotion, and use all 10 classes in CLINC150 as there are out-of-distribution examples available in the training split. We check how well q_{GFN} manages to identify which are in-distribution classes and which are out-of-distribution while further performing fine-grained classification for in-distribution classes. While MultiNERD is a comparatively easy task, GoEmotion and CLINC150 are much harder tasks (from a classification point of view) – for instance, a BERT-based (Devlin et al., 2019) classifier trained on the complete training set (unlike the few-shot setup in our approach) scores an average f1-score of 0.65 ± 0.13 for our sampled classes for GoEmotion. We also provide test set embeddings reduced to two dimensions using t-sne (van der Maaten and Hinton, 2008) and Roberta-large (Liu et al., 2019) in Figure 10 – note how there are no clear, well-defined clusters for GoEmotion or CLINC150 while the training distribution for MultiNERD is clearly defined into separate clusters.

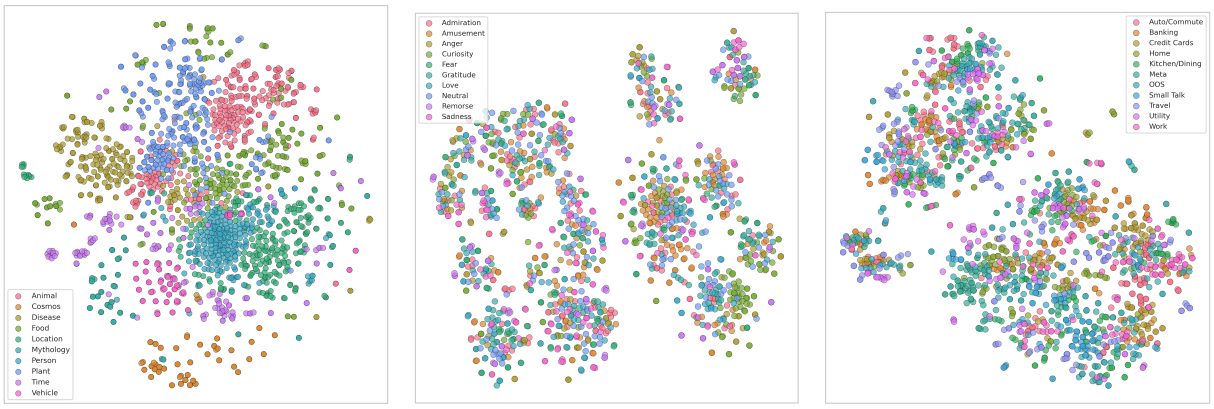
A.7 Training details

Gaussian Process We fit a Gaussian Process (Rasmussen, 2004) to to model out-of-distribution log probabilities (after applying the logit normalisation trick) δ for class c using a linear mean and a Matérn kernel (with $\nu = 5/2$) (Genton, 2002) defined as

$$f_c(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}),\ k(\mathbf{x}, \mathbf{x}'))$$
$$m(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$
$$k(\mathbf{x}, \mathbf{x}') = \sigma^2(1 + \sqrt{5}r/\ell + 5r^2/3\ell^2)\exp(-\sqrt{5}r/\ell)$$

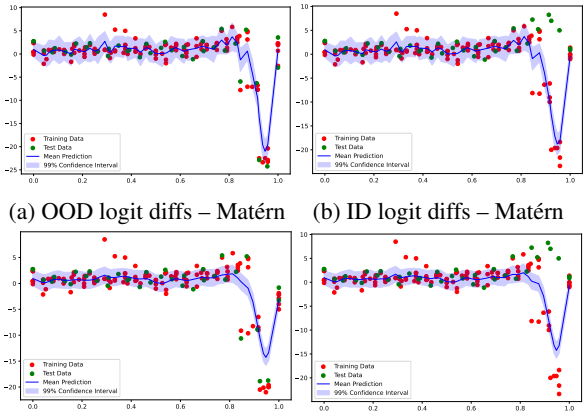
where $\ell, \sigma, \mathbf{w}, b$ are trainable parameters and $r = \|\mathbf{x} - \mathbf{x}'\|_2$. We decided to use the Matérn kernel ($\nu = 5/2$) over the RBF kernel – though the latter is simpler, it was not able to fit logit differences as well as the former in the same number of optimiser steps (check Figure 11). We use the Huggingface transformers library (Wolf et al., 2020) for loading the base models, tokenisers and datasets.

We perform extensive hyperparameter sweeps for a range of hyperparameters which we list in Table 4. Note that where we mention ranges with an asterisk(*), we pick the best hyperparameter from these ranges for a particular support set and choice



(a) MultiNERD (b) GoEmotion (c) CLINC150

Figure 10: Visualitions of t-SNE (van der Maaten and Hinton, 2008) test embeddings using Roberta-large (Liu et al., 2019) which highlights the complexity of GoEmotion (Demszky et al., 2020) and CLINC150 (Larson et al., 2019).



(a) OOD logit diffs – Matérn (b) ID logit diffs – Matérn

(c) OOD logit diffs – RBF (d) ID logit diffs – RBF

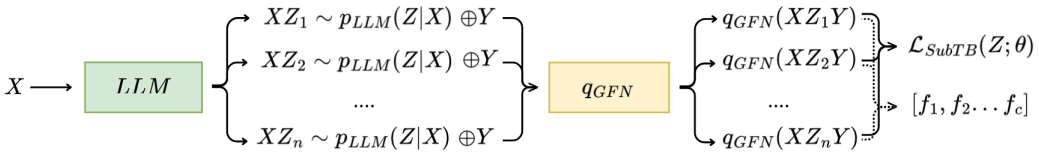
Figure 11: Fitting out-of-distribution and in-distribution logit differences with the RBF and Matérn kernels.

of $K \in \{4, 8, 16\}$. We also train q_{GFN} for a maximum of the mentioned epochs and select the model which performs best on the validation set. We also ensure that the total latents (support and query) add up to exactly 72. We perform validation after every 15 episodes. We use AdamW as our optimiser for both q_{GFN} and use a learning rate of $1e-1$ to train all GPs. We use FlashAttention-2 (Dao, 2023) for all LLMs. We also use ChatGPT (`https://openai.com/index/chatgpt/`) as a coding assistant for writing small snippets in our work. We use a single NVIDIA A100 GPU with 40 GB memory and each model takes between 30 minutes to 24 GPU hours to train.

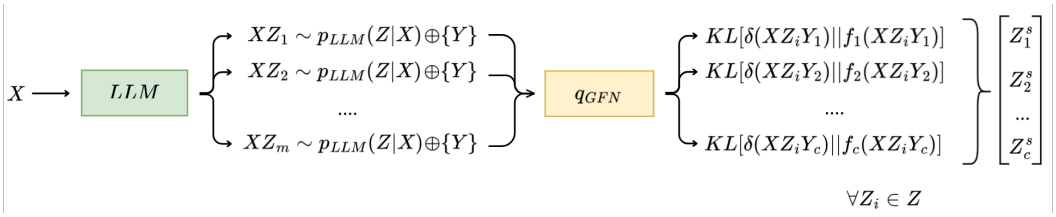
Training with reward buffers We can train q_{GFN} with reward buffers which cache $R(XZY)\ \forall\ XZY$ during the first epoch which speeds up our training significantly – we find that training time for subsequent epochs falls by almost $\sim 40\%$ which is particularly beneficial for training higher values of $\{N, K\}$. For example, when we have $\{N, K\} = \{5, 16\}$ for MultiNERD, training q_{GFN} for the first epoch takes $\sim 24,000$ seconds but the subsequent epochs take only $\sim 14,500$ seconds each. Note that we do not perform "on-policy" training (sampling latents depending on X directly in the training flow). We depict our complete training and testing flow diagrammatically in Figure 12 and describe our testing method with q_{GFN} in Algorithm 2, having previously described our training flow in Algorithm 1.

A.8 Ablation studies

Evaluating W-GFN (3B) with CoT$^-$ (70B) As demonstrated previously, adding an out-of-distribution hint in the input prompt increases performance for CoT reasoning for Llama-3.2 3B. This is also true for Llama-3.3 70B, interestingly, as it is expected that language models become better reasoners with scale (Kojima, 2024). From Table 5, it is demonstrable that W-GFN, with only 3B parameters, is able to identify out-of-distribution classes without the need of an explicit hint - further reinforcing that it learns a true representation of the training distribution using latent variables rather than rely on extensive prompt engineering. For simpler tasks such as MultiNERD, where the out-of-distribution example is more evident and less obvious, Llama-3.3 70B performs better than W-GFN, albeit not by a large margin (W-GFN matches around 91.13% of the seen setting performance and 80.80% of the unseen setting performance of CoT



(a) Training W-GFN. We minimise $\mathcal{L}_{SubTB}(Z;\theta)$ and learn $[f_1..f_c]$ to predict out-of-distribution signals.



(b) Testing W-GFN. $[Z_1^s, Z_2^s ... Z_c^s]$ represents the normalised Z-Scores of KL div between logit normalised values and f_c.

Figure 12: Training and testing for W-GFN

$X \rightarrow$ ...Which class does the entity ['endemic'] in the sentence 'In another study, it reduced the number of symptomatic cases after exposure to leptospirosis under heavy rainfall in endemic areas.' represent?

$Z|X \rightarrow$ The test entity broadly represents a geographical location or region where a disease is consistently present and prevalent.

Figure 13: The sampled latent refers to classes *Location* and *Disease* which can lead to a misclassification by q_{GFN} when it is asked to distinguish between the two. "..." refers to few-shot demonstrations in the support set.

Support Classes	CoT (3B)	W-GFN (3B)	CoT$^-$(70B)	CoT (70B)
Fear, Love, Gratitude	$0.31_{0.02}$ $0.40_{0.02}$	$0.52_{0.01}$ $0.60_{0.08}$	$0.48_{0.02}$ $0.25_{0.03}$	$0.63_{0.02}$ $0.76_{0.02}$
Admiration, Gratitude, Anger, Fear	$0.34_{0.02}$ $0.29_{0.03}$	$0.56_{0.02}$ $0.50_{0.04}$	$0.52_{0.03}$ $0.15_{0.02}$	$0.57_{0.03}$ $0.51_{0.02}$
Remorse, Love, Sadness, Admiration, Gratitude	$0.29_{0.03}$ $0.21_{0.05}$	$0.53_{0.05}$ $0.63_{0.06}$	$0.62_{0.02}$ $0.37_{0.08}$	$0.66_{0.03}$ $0.76_{0.02}$
Disease, Food, Cosmos, Location	$0.54_{0.01}$ $0.36_{0.04}$	$0.72_{0.01}$ $0.55_{0.05}$	$0.79_{0.03}$ $0.68_{0.01}$	$0.90_{0.02}$ $0.93_{0.03}$
Banking, Credit Cards, Dining, Home, Auto/Commute, Travel, Utility, Work, Small Talk, Meta	$0.47_{0.02}$ $0.21_{0.03}$	$0.55_{0.02}$ $0.30_{0.01}$	$0.78_{0.01}$ $0.17_{0.01}$	$0.80_{0.01}$ $0.45_{0.05}$

Table 5: Results with Llama-3.3 70B

with Llama-3.3 70B) considering the overall size difference and richness of pre-training tasks.

Evaluating CoT$^-$(3B) CoT$^-$(3B) refers to the baseline using Llama-3.2 3B without providing an out-of-distribution or unseen class hint in the input prompt. We almost universally observe lower performance for this setting as opposed to providing a hint - indicating that smaller LLMs generally fail to detect unseen classes in the open-set classification setting natively and need to be explicitly prompted to do so. For CLINC150, the difference is even more striking as we observe that with a higher number of classes and noisy few-shot demonstrations, the model fails to robustly detect out-of-distribution test examples. We present these results for the tasks MultiNERD, GoEmotion and CLINC150 in Table 6, Table 7 and Table 9 respectively. Similar to CoT$^-$(70B), there is a massive drop in performance once the out-of-distribution hint is removed which underscores the dependence of generative LLMs on user prompts.

Using non-anonymised labels We conduct an additional ablation study on our datasets, evaluating model performance when labels are not anonymised (i.e., original labels). In this setup, models are evaluated on their ability to explicitly return either a correct (non-anonymised) support label or "None of these" for unseen classes. Interestingly, we see from Table 8 that W-GFN continues to outperform all baselines in the datasets MultiNERD and GoEmotion. This also shows that anonymised evaluation is a special case of a more general and challenging classification setting as generally non-anonymised scores are higher than the anonymised scores due to semantic information leaking from the non-anonymised labels. In particular, note the steadily decreasing scores of *Fear, Love, Gratitude* with increasing K – a phenomenon attributed to increasing memorisation with noisier few-shot examples (we noticed the model tended

	K=4		K=8		K=16	
Support Classes	**CoT$^-$**	**CoT**	**CoT$^-$**	**CoT**	**CoT$^-$**	**CoT**
Cosmos, Food, Mythology	$0.37_{0.03}$ $0.32_{0.02}$	$0.37_{0.03}$ $0.54_{0.03}$	$0.39_{0.02}$ $0.35_{0.03}$	$0.40_{0.01}$ $0.55_{0.01}$	$0.39_{0.02}$ $0.31_{0.01}$	$0.41_{0.04}$ $0.49_{0.01}$
Person, Plant, Vehicle	$0.32_{0.03}$ $0.29_{0.05}$	$0.32_{0.03}$ $0.64_{0.03}$	$0.31_{0.02}$ $0.35_{0.03}$	$0.34_{0.03}$ $0.58_{0.02}$	$0.32_{0.02}$ $0.29_{0.01}$	$0.38_{0.05}$ $0.60_{0.01}$
Location, Disease, Time	$0.37_{0.02}$ $0.06_{0.02}$	$0.40_{0.00}$ $0.35_{0.03}$	$0.39_{0.02}$ $0.13_{0.02}$	$0.39_{0.02}$ $0.45_{0.03}$	$0.37_{0.01}$ $0.17_{0.03}$	$0.39_{0.02}$ $0.39_{0.03}$
Disease, Food, Cosmos, Location	$0.47_{0.02}$ $0.13_{0.01}$	$0.47_{0.04}$ $0.47_{0.03}$	$0.51_{0.03}$ $0.09_{0.02}$	$0.54_{0.01}$ $0.36_{0.04}$	$0.45_{0.03}$ $0.16_{0.02}$	$0.45_{0.04}$ $0.33_{0.02}$
Person, Plant, Time, Mythology	$0.37_{0.01}$ $0.25_{0.04}$	$0.38_{0.02}$ $0.47_{0.03}$	$0.38_{0.01}$ $0.18_{0.02}$	$0.39_{0.03}$ $0.40_{0.04}$	$0.37_{0.02}$ $0.20_{0.02}$	$0.43_{0.04}$ $0.37_{0.01}$
Plant, Animal, Mythology, Disease	$0.38_{0.03}$ $0.37_{0.03}$	$0.39_{0.02}$ $0.56_{0.04}$	$0.4_{0.03}$ $0.46_{0.02}$	$0.40_{0.03}$ $0.65_{0.02}$	$0.35_{0.02}$ $0.44_{0.04}$	$0.35_{0.02}$ $0.60_{0.01}$
Animal, Person, Time, Plant, Disease	$0.52_{0.01}$ $0.28_{0.02}$	$0.46_{0.04}$ $0.52_{0.05}$	$0.45_{0.02}$ $0.36_{0.05}$	$0.43_{0.03}$ $0.47_{0.04}$	$0.41_{0.02}$ $0.3_{0.02}$	$0.41_{0.02}$ $0.44_{0.03}$
Plant, Location, Vehicle, Disease, Food	$0.44_{0.02}$ $0.14_{0.06}$	$0.42_{0.04}$ $0.39_{0.01}$	$0.45_{0.01}$ $0.29_{0.02}$	$0.41_{0.08}$ $0.52_{0.03}$	$0.47_{0.02}$ $0.16_{0.03}$	$0.50_{0.03}$ $0.42_{0.01}$
Animal, Mythology, Food, Location, Person	$0.42_{0.01}$ $0.18_{0.02}$	$0.37_{0.03}$ $0.45_{0.05}$	$0.38_{0.05}$ $0.17_{0.01}$	$0.39_{0.02}$ $0.41_{0.03}$	$0.40_{0.01}$ $0.18_{0.03}$	$0.42_{0.02}$ $0.38_{0.02}$

Table 6: Chain-of-thought reasoning for NERD (using anonymised labels) without (denoted by CoT$^-$) and with (denoted by CoT) providing a hint that an example might belong to an unseen class using Llama-3.2 3B.

	K=4		K=8		K=16	
Support Classes	**CoT$^-$**	**CoT**	**CoT$^-$**	**CoT**	**CoT$^-$**	**CoT**
Fear, Love, Gratitude	$0.31_{0.01}$ $0.23_{0.02}$	$0.34_{0.01}$ $0.59_{0.03}$	$0.29_{0.04}$ $0.10_{0.02}$	$0.31_{0.02}$ $0.40_{0.02}$	$0.25_{0.02}$ $0.08_{0.01}$	$0.27_{0.01}$ $0.10_{0.01}$
Curiosity, Remorse, Amusement	$0.22_{0.03}$ $0.19_{0.02}$	$0.24_{0.03}$ $0.40_{0.03}$	$0.22_{0.01}$ $0.13_{0.01}$	$0.23_{0.02}$ $0.23_{0.02}$	$0.21_{0.01}$ $0.08_{0.01}$	$0.23_{0.02}$ $0.10_{0.02}$
Gratitude, Anger, Remorse	$0.27_{0.01}$ $0.35_{0.05}$	$0.33_{0.01}$ $0.49_{0.03}$	$0.31_{0.02}$ $0.39_{0.01}$	$0.32_{0.02}$ $0.44_{0.01}$	$0.22_{0.02}$ $0.35_{0.05}$	$0.32_{0.03}$ $0.29_{0.02}$
Admiration, Gratitude, Anger, Fear	$0.31_{0.03}$ $0.09_{0.03}$	$0.27_{0.01}$ $0.31_{0.03}$	$0.35_{0.02}$ $0.11_{0.02}$	$0.34_{0.02}$ $0.29_{0.03}$	$0.28_{0.01}$ $0.09_{0.01}$	$0.36_{0.02}$ $0.21_{0.09}$
Fear, Remorse, Love, Admiration	$0.25_{0.02}$ $0.18_{0.02}$	$0.28_{0.01}$ $0.46_{0.06}$	$0.24_{0.04}$ $0.14_{0.02}$	$0.31_{0.01}$ $0.24_{0.01}$	$0.17_{0.02}$ $0.12_{0.02}$	$0.23_{0.01}$ $0.27_{0.04}$
Love, Amusement, Curiosity, Sadness	$0.26_{0.02}$ $0.13_{0.03}$	$0.29_{0.05}$ $0.38_{0.04}$	$0.24_{0.01}$ $0.15_{0.02}$	$0.24_{0.02}$ $0.23_{0.01}$	$0.23_{0.02}$ $0.18_{0.02}$	$0.28_{0.02}$ $0.11_{0.03}$
Fear, Remorse, Admiration, Curiosity, Anger	$0.28_{0.02}$ $0.13_{0.03}$	$0.32_{0.03}$ $0.35_{0.03}$	$0.18_{0.01}$ $0.17_{0.04}$	$0.28_{0.00}$ $0.13_{0.03}$	$0.17_{0.01}$ $0.15_{0.02}$	$0.27_{0.02}$ $0.11_{0.02}$
Love, Anger, Gratitude, Curiosity, Amusement	$0.21_{0.02}$ $0.12_{0.02}$	$0.29_{0.02}$ $0.38_{0.03}$	$0.23_{0.01}$ $0.14_{0.02}$	$0.29_{0.01}$ $0.17_{0.02}$	$0.14_{0.03}$ $0.15_{0.03}$	$0.30_{0.01}$ $0.16_{0.03}$
Remorse, Love, Sadness, Admiration, Gratitude	$0.26_{0.02}$ $0.13_{0.03}$	$0.23_{0.02}$ $0.31_{0.01}$	$0.21_{0.04}$ $0.14_{0.03}$	$0.29_{0.03}$ $0.21_{0.05}$	$0.14_{0.03}$ $0.09_{0.03}$	$0.28_{0.02}$ $0.20_{0.03}$

Table 7: Chain-of-thought reasoning for GoEmotion (using anonymised labels) without (denoted by CoT$^-$) and with (denoted by CoT) providing a hint that an example might belong to an unseen class using Llama-3.2 3B.

	K = 4			K = 8			K = 16		
Support Classes	**SFT**	**CoT**	**W-GFN**	**SFT**	**CoT**	**W-GFN**	**SFT**	**CoT**	**W-GFN**
Person, Plant, Vehicle	$0.57_{0.02}$ $0.00_{0.00}$	$0.75_{0.02}$ $0.40_{0.03}$	$\mathbf{0.80_{0.03}}$ $\mathbf{0.88_{0.02}}$	$0.72_{0.03}$ $0.00_{0.00}$	$0.75_{0.03}$ $0.47_{0.04}$	$\mathbf{0.76_{0.02}}$ $\mathbf{0.88_{0.01}}$	$0.65_{0.01}$ $0.00_{0.00}$	$\mathbf{0.80_{0.03}}$ $0.37_{0.04}$	$\mathbf{0.80_{0.01}}$ $\mathbf{0.86_{0.02}}$
Fear, Love, Gratitude	$0.50_{0.01}$ $0.01_{0.01}$	$0.46_{0.02}$ $0.48_{0.02}$	$\mathbf{0.58_{0.04}}$ $\mathbf{0.57_{0.01}}$	$0.51_{0.05}$ $0.02_{0.02}$	$0.51_{0.03}$ $0.30_{0.02}$	$\mathbf{0.61_{0.01}}$ $\mathbf{0.58_{0.02}}$	$0.46_{0.02}$ $0.04_{0.02}$	$0.49_{0.03}$ $0.17_{0.02}$	$\mathbf{0.61_{0.01}}$ $\mathbf{0.66_{0.06}}$

Table 8: Using non-anonymised labels for all baselines with using Llama-3.2 3B.

to not follow instructions and directly reply with a semantically extrapolated non-anonymised class - for example, instead of recognising an emotion as out-of-distribution, it returned an answer of *Astonishment/Surprise* etc - which points to overfitting on pre-training data rather than reasoning on a downstream task).

Impact of using a Wasserstein offset One possible explanation for more stable training with the Wasserstein offset can be that gradually adding a decreasing low reward offset encourages q_{GFN} to explore earlier transitions with rewards not as significantly different as those associated with a much lower reward gotten after adding a high, negative reward offset. This makes it more capable of general-

	K=4		K=8		K=16	
Support Classes	**CoT$^-$**	**CoT**	**CoT$^-$**	**CoT**	**CoT$^-$**	**CoT**
Banking, Credit Cards, Dining, Home, Auto/Commute, Travel, Utility, Work, Small Talk, Meta	$0.43_{0.01}$ $0.00_{0.00}$	$0.41_{0.01}$ $0.25_{0.02}$	$0.49_{0.01}$ $0.00_{0.00}$	$0.47_{0.02}$ $0.21_{0.03}$	$0.52_{0.01}$ $0.00_{0.00}$	$0.52_{0.02}$ $0.19_{0.02}$

Table 9: Chain-of-thought reasoning for CLINC150 (using anonymised labels) without (denoted by CoT$^-$) and with (denoted by CoT) providing a hint that an example might belong to an unseen class using Llama-3.2 3B. The last row denotes the average score across the seen and unseen setting.

isation on state transitions outside the training-time transitions provided and does not leave "gaps" in the transition reward landscape. Note that treating the reward offset as a hyperparameter (we used the value $\Delta = -80$) can yield better results – which is an avenue we did not explore and can be of further interest to the research community. Another alternative can be using a learnable reward offset, but that would require a learning objective modelled into the subtrajectory balance loss.

Reward landscapes for latent sequences From the subtrajectory loss balance in Equation 1, we depict the term denoting the difference of rewards given as

$$\log \Delta R = \log \frac{R(z_{1:i}\top)}{R(z_{1:j}\top)}$$
$$= \log R(z_{1:i}\top) - \log R(z_{1:j}\top) \; \forall \; \{i, j\}$$

in Figure 14 for very long sequences Z. After normalising by length of the total sequence XZY, note that the overall similarity of the reward landscape is very similar for these sequences – which leads us to believe that a possible further area of research in tasks requiring long chains of thought would be to model this reward landscape using a few sequences using a Gaussian Process and then sampling from the unnormalised surrogate reward model which would be computationally faster as well as cheaper than computing the reward signal from the pretrained LLM (note that we do not need the exact log-probabilities – we just need state transition probabilities to follow the rules of the GFN). We leave testing this idea to future work in the area.

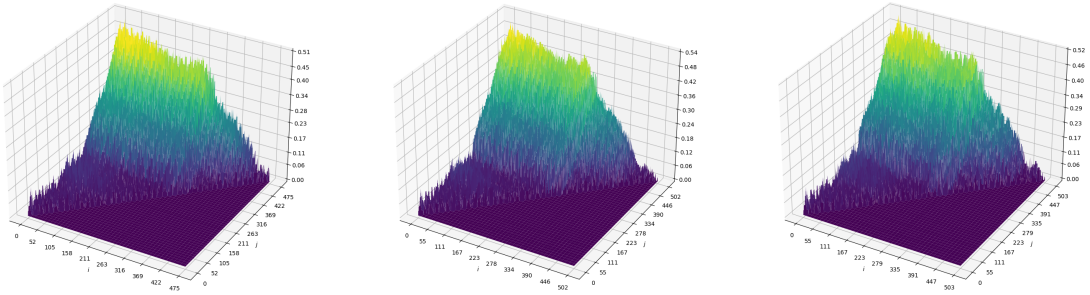
Areas of further research Some future research directions to improve reasoning using W-GFNs for sparse-shot learning and other general tasks are:

- The hidden variable or conditional latent $Z|X$ is the most important factor for training q_{GFN} as it enables the model to explore similar hidden variables. From our analysis, the biggest reason for within-distribution or in-distribution/out-of-distribution misclassifications were latent variables which contained information about multiple classes (see Figure 13 for an example). Intelligently sampling $Z|X$ while maintaining diversity can thus lead to a tremendous improvement in performance.

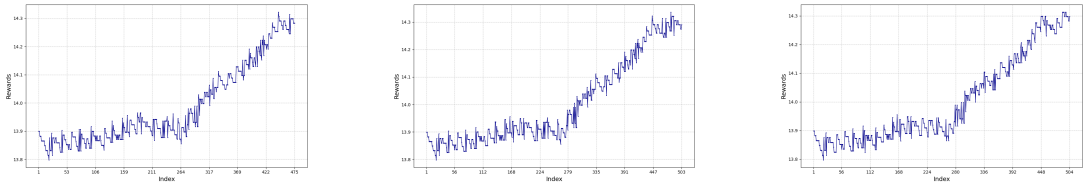
- We directly train the posterior as $p(Y|X) \propto \Sigma_Z p(Y|X, Z)p(Z|X) \approx \Sigma_Z q_{GFN}(Y|X, Z)$ and forego training with a prior function as we assume that all latents $Z|X$ are equally likely and distributed uniformly. This works reasonably well, however, performance can be enhanced by defining a prior function $p(Z|X)$ which can "weigh" latents and thus prioritise latents which are more descriptive of the query in X. A possible way to achieve this is to use a similarity-based encoder model such as RoBERTa (Liu et al., 2019), generative LLM approaches such as CoT-decoding (Wang and Zhou, 2024) to measure relevance etc.

- Similarly, to improve performance at inference, we can apply a form of Bayesian model averaging wherein we prioritise latents which are more probable to describe the test example by assigning them a higher prior probability. This has an advantage over uniformly weighted model averaging (which we adopt currently) as less likely latents would ideally have lower priority.

- We use a simple heuristic for detecting out-of-distribution signals (a median centralised Z-score), however, this can be extended to a probabilistic measure where we explicitly model the posterior probability of log probabilities being out-of-distribution and learn a more accurate cut-off threshold (train a classifier, for example) to further increase accuracy.



(a) Landscape of $\log \Delta R_1$ (b) Landscape of $\log \Delta R_2$ (c) Landscape of $\log \Delta R_3$



(d) $\log \Delta R_1$ (e) $\log \Delta R_2$ (f) $\log \Delta R_3$

Figure 14: Even for very long sequences, $\log \Delta R$ on the z-axis has a somewhat consistent reward landscape