Life of Research · Published 23 November 2026 · DOI 10.5281/zenodo.24450652

A Composed Policy Is Not a Predicted Run: What Combining Policies Trained on Other Rewards Can Tell About Reinforcement Learning on a New One, Why the Closed Form Covers an Objective Common Recipes Do Not Optimise, and the Null Basis That Would Separate Reward-Specific Prediction From Shared Drift

Pranay Mahendrakar
SONYTECH · ORCID 0009-0003-7224-029X
pranaymahendrakar@sonytech.in · mahendrakarpranay@gmail.com · +91 6361723454

Abstract. Post-training a language model with reinforcement learning is expensive and must be repeated whenever the reward changes. A September 2026 proposal, PoEM, argues that the outcome of such training on a new reward can be predicted without running it, by composing the log-probabilities of policies already trained on other rewards, and reports that these policies span a low-dimensional subspace of log-likelihood space even when their weight updates are nearly orthogonal. In this analysis the author reviews and synthesises published results to show that the composition is exact for one object, the KL-regularised optimum of a linear combination of rewards, and that the policy a training recipe actually produces departs from that object in five documented ways: reward scale discarded by group normalisation and by preference losses, KL penalties implemented in forms whose gradient is disputed or omitted altogether, offline implicit rewards that generalise worse than explicit ones, token-level normalisation where the closed form is sequence-level, and target runs chosen to exclude reward hacking. The paper argues that the reported low rank is measured mostly on responses sampled from the reference model; on the experts' own responses, PoEM's own appendix shows the gap between reward rank and log-ratio rank narrowing on one basis and reversing on another. A growing literature finds that a large part of what reinforcement learning does to a language model is independent of the reward: on reasoning benchmarks, and for some model families, random rewards, entropy minimisation and training-free sharpening of the base model each recover much of the gain. PoEM's own baselines are consistent with this reading: on ten held-out reward models, uniform weights that never see the target reward recover 0.50 of the gain against 0.56 for fitted weights. The paper consolidates 30 published results in a table and a figure, separates three things "the RL outcome" can mean, states a seven-step procedure for reading a composed-policy forecast, and proposes a null basis of experts trained on uninformative rewards as the control that would separate reward-specific prediction from shared drift.

Keywords: language-model post-training; KL-regularised policy optimisation; policy composition; multi-objective alignment; decoding-time alignment; implicit reward; low-rank structure; evaluation methodology

1. Introduction

Reinforcement learning from feedback is now a routine stage in building a language model. A reward, learned from human comparisons or computed by a program, scores sampled responses, and a policy-gradient method moves the model toward responses that score well while a penalty keeps it near its starting distribution (Christiano et al., 2017; Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022). The stage is costly, it is sometimes unstable, and its result is specific to the reward used. Change the reward model, add a second objective, or reweight two existing ones, and the standard answer is to run the stage again.

A line of work going back at least to the soft-optimality view of reinforcement learning offers a shortcut. When the training objective is expected reward minus a scaled KL divergence from a reference policy, its optimum has a closed form: the reference distribution tilted by the exponentiated reward (Peng et al., 2019; Levine, 2018; Korbak et al., 2022; Rafailov et al., 2023). Because the tilt is exponential in the reward, a weighted sum of rewards produces an optimum whose log-probability is the same weighted sum of the individual optima's log-ratios to the reference. If the policies trained on each reward were the exact optima, the policy for any linear combination of those rewards could be written down without training.

Hamidieh, Daras and Torralba (2026) build a method, PoEM, on this observation and push it further. They train bases of up to twenty adapters on Qwen3-0.6B, one per reward, and ask whether composing them predicts what reinforcement learning on a new reward would produce. For rewards that are exact weighted sums of the basis rewards, they report that the composed policy recovers most of the reward gain of a directly trained policy, with an error close to the difference between two training runs. For rewards that are not such sums, they report a geometric finding: the experts' weight updates are nearly orthogonal and close to full rank, yet the matrix of their log-ratios to the reference model has an effective rank of

© 2026 Pranay Mahendrakar. Licensed under CC BY 4.0.

6.3 out of 20. They propose a coverage score, the held-out variance of the new reward explained by the experts' log-ratios, as a way to tell in advance which new rewards the basis can reach. The paper is careful: it states that it has only checked small models, that exact composition assumes KL-regularised optima, and that many post-training runs omit the KL term. Its title and abstract nonetheless make a broader claim, that reinforcement learning outcomes can be predicted from existing policies, and the claim deserves to be read against the wider record.

That record pulls in two directions. One body of work supports composition. Weight interpolation between policies fine-tuned on different rewards traces a front close to the one obtained by retraining on each weighting (Rame et al., 2023); mixing next-token predictions under the closed form outperforms parameter merging and comes with optimality guarantees in a tabular setting (Shi et al., 2024); applying the difference between a small tuned and untuned model to a large untuned model recovers most of the gap to the large model's tuned version (Liu, A. et al., 2024; Mitchell et al., 2024). The older reinforcement-learning literature on transfer and composition gives conditions under which combining value functions or policies trained on component tasks yields an optimal or near-optimal policy for a new task (Barreto et al., 2017; Haarnoja et al., 2018; Hunt et al., 2019; Nangue Tasse et al., 2020).

A second body of work, mostly from 2025 and 2026, describes what reinforcement learning on language models actually does, and much of it is not about the reward. Training with random rewards raises one model family's math accuracy almost as much as training with correct ones (Shao, R. et al., 2025). Rewarding the model for its own confidence, or for low entropy, matches standard reward-driven training on several reasoning benchmarks (Zhao et al., 2025; Agarwal et al., 2025). Sampling from a sharpened version of the base model, with no training at all, nearly matches reinforcement learning on several single-shot tasks (Karan and Du, 2025). Reinforcement learning updates touch a small subset of parameters whose overlap across seeds, data and algorithms exceeds chance (Mukherjee et al., 2025), concentrate in regions that are consistent across datasets and recipes (Zhu et al., 2025), and are dominated by a rank-one direction per weight matrix that evolves linearly (Cai et al., 2025). If a large share of what any run does is common to every reward under a given recipe, a composition of runs will reproduce that share for any new reward, and some of what looks like prediction will be the shared part.

The question this paper addresses is what a composed policy licenses as a forecast of reinforcement learning on a new reward. In this analysis the author reviews and synthesises published results to show three things. First, "the RL outcome" names at least three different targets, and the closed form is exact for only one of them. Second, the policies that common recipes produce depart from that target in five documented ways, and PoEM's own results track those departures: the bases whose training most closely matches the closed form's objective need the least correction. Third, the low-rank finding and the held-out successes are both consistent with a reward-independent drift that every expert shares, and PoEM's own baselines show most of the held-out recovery to be available without fitting weights to the target. None of this shows that composition fails. It shows that the evidence so far measures a weaker property than prediction, and it identifies the control that would measure the stronger one.

Section 2 sets out the closed form and the two literatures that build on it. Section 3 separates three meanings of the outcome. Section 4 takes the five departures in turn. Section 5 asks which of several published low-rank findings PoEM's is. Section 6 states the shared-drift reading and tests it against PoEM's baselines. Section 7 compares composition in weight space and in policy space. Section 8 consolidates the measured values in Table 1 and Figure 1. Section 9 states a reading procedure (Algorithm 1). Sections 10 to 12 give the strongest objection, what is not known, and the studies that would settle it.

2. The Closed Form and Its Lineage

2.1 The tilted reference

Write pi_ref for the reference policy, r for a reward on prompt-response pairs, and beta for the strength of a KL penalty. The objective is the expected reward under the policy minus beta times the KL divergence from the policy to pi_ref, averaged over prompts. Over an unrestricted policy class its maximiser assigns each response a probability proportional to its reference probability times exp(r/beta). This identity appears under several names: as the solution of relative-entropy policy search and advantage-weighted regression (Peng et al., 2019), as posterior inference in the control-as-inference framing (Levine, 2018), and, for language models specifically, as the target distribution of variational inference with the reference model as a prior (Korbak et al., 2022). Direct preference optimisation rearranges it to express the reward through the policy, so that beta times the log-ratio of a trained policy to the reference is the reward for which that policy is optimal, up to a term that depends only on the prompt (Rafailov et al., 2023). KL control toward a pretrained prior was used in dialogue before preference-based fine-tuning of large models became standard (Jaques et al., 2019), and early applications of preference learning to language models put the same penalty into the reward (Ziegler et al., 2019; Stiennon et al., 2020).

Two consequences follow directly. If r is a weighted sum of component rewards and each component policy is the exact optimum for its own reward at a shared beta and

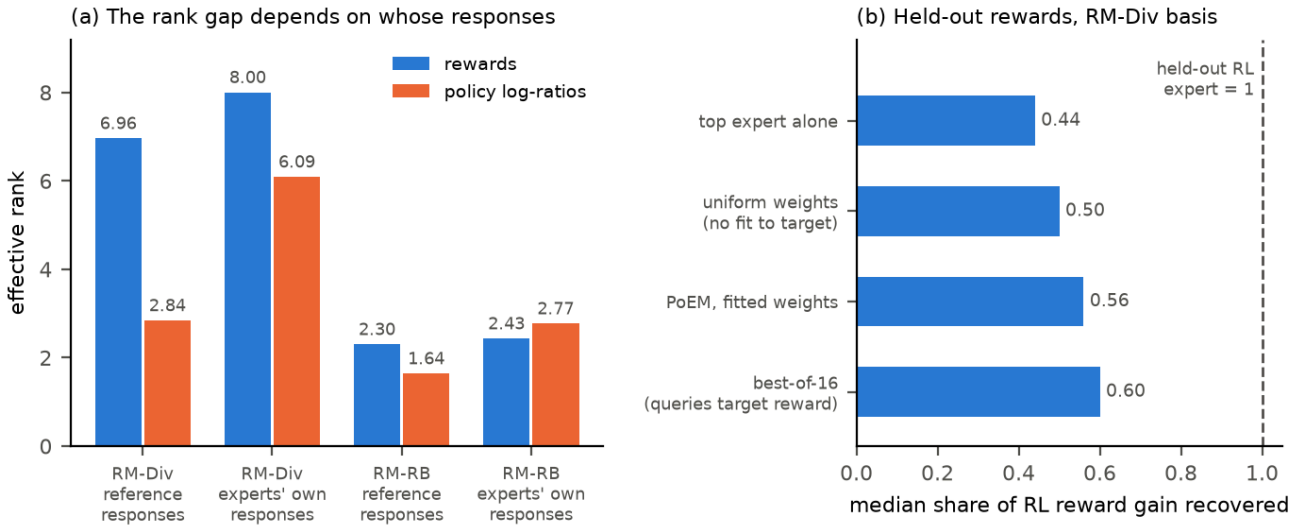


Figure 1. (a) Effective rank of the matrix of rewards (blue) and of policy log-ratios to the reference (orange) for the ten-member RM-Div basis and the four-member RM-RB basis, measured on reference-model responses and on the experts' own responses. (b) Median share of the held-out expert's reward gain recovered on the ten held-out RM-Div rewards by the top single expert, by uniform weights that do not use the target reward, by PoEM with weights fitted to the target reward, and by best-of-16 sampling under the target reward; the dashed line marks the held-out RL-trained expert. All values are as published by Hamidieh et al. (2026): panel (a) from their Appendix C.5, panel (b) from their Section 4.3.

reference, then the optimum for the sum is the reference times the product of each component's ratio to the reference, raised to its weight. And beta times each trained policy's log-ratio is an implicit reward. PoEM uses both: the first to compose experts when the weights are known, the second to estimate the weights by regressing the new reward on the experts' log-ratios over a small calibration set (Hamidieh et al., 2026).

2.2 Composition in reinforcement learning before language models

The idea that policies trained on one set of rewards can be reused for another is older than language-model post-training. Successor features separate a value function into features of the environment's dynamics and a reward-weight vector, so that for any reward linear in those features the value of every stored policy can be computed without learning; generalised policy improvement then acts on the best of them, with performance guarantees for the transferred policy (Barreto et al., 2017). The guarantee is on policy evaluation for linear rewards; the composed policy is improved over every stored policy, not proved optimal for the new reward.

For entropy-regularised reinforcement learning, which is the setting of the closed form, composition has a known gap. Haarnoja et al. (2018) showed that adding the soft Q-functions of policies trained on two rewards gives a policy for the summed reward whose suboptimality is bounded by a divergence between the composed policies. Hunt et al. (2019) analysed that construction and the successor-feature one, showed that each performs poorly in some situations, and proposed learning a correction term, the discounted future divergence between base policies, which in principle recovers the optimal composed policy. Nangue Tasse et al. (2020) proved that for goal-reaching tasks with restricted dynamics, specific compositions of goal-oriented value functions recover the optimal policy for every task expressible in a Boolean algebra over the base tasks. The lesson of this literature is that exactness holds under stated conditions on rewards and dynamics, and that outside them the composed policy carries an error whose size depends on how much the component policies disagree.

2.3 Composition at decoding time

For language models the composed policy is usually built at decoding time from next-token distributions. DExperts multiplies a base model's distribution by an expert's and divides by an anti-expert's (Liu, A. et al., 2021); contrastive decoding scores tokens by the gap between a large and a small model's log-likelihoods (Li, X. L. et al., 2023). Emulated fine-tuning and proxy-tuning use the closed form explicitly: the log-ratio between a small tuned model and its untuned base is treated as an implicit reward and added to a larger untuned model's log-probabilities (Mitchell et al., 2024; Liu, A. et al., 2024). Proxy-tuning reports closing 88% of the gap between Llama2-70B and its chat-tuned version using 7B proxies (Liu, A. et al., 2024). Decoding-time realignment interpolates and extrapolates between an aligned model and its reference in log space to emulate different KL strengths without retraining (Liu, T. et al., 2024); model extrapolation does the analogous thing in weight space (Zheng et al., 2025).

Multi-objective decoding (MOD) is the closest prior method to PoEM, which says so (Hamidieh et al., 2026). MOD derives, for a family of f-divergence-regularised objectives, a closed-form optimum for any weighting of objectives and decodes it token by token; for the reverse KL used by PPO and DPO the rule reduces to a weighted product of the component policies, that is, a linear combination of their logits (Shi et al., 2024). MOD reports a 12.8% relative improvement in weighted reward over parameter merging when three objectives on a helpful-assistant task are weighted equally, and it states two limitations: its analysis is tabular and ignores function approximation, and decoding a response-level distribution at the token level may degrade performance. PoEM's own appendix notes that MOD's reverse-KL rule coincides with PoEM at its default strength when the weights are given (Hamidieh et al., 2026).

Other decoding-time methods replace the trained policies with value estimates. Controlled decoding trains a prefix scorer for a reward and uses it to steer a frozen model, and reports that scorers for different rewards can be combined at inference to solve a multi-objective problem without further training (Mudgal et al., 2024). Value augmented sampling learns a value function from samples of the frozen model alone and likewise composes rewards at deployment (Han et al., 2024). Transfer Q-star estimates the optimal value for a target reward through a model aligned to a different baseline reward (Chakraborty et al., 2024). ARGS adds a reward-model score to the next-token logits (Khanov et al., 2024); GenARM trains an autoregressive reward model so that next-token rewards are defined (Xu, Y. et al., 2025); linear alignment estimates the aligned response in closed form in a single inference step (Gao, S. et al., 2024); weak-to-strong search maximises a small model pair's log-likelihood gap while sampling from a large model (Zhou, Z. et al., 2024a). Model arithmetic gives a general algebra for composing models at decoding time (Dekoninck et al., 2024).

2.4 Composition in weight space and in training

Model soups average the weights of models fine-tuned from one pretrained initialisation with different hyperparameters, on the observation that such models often sit in a single low-error basin (Wortsman et al., 2022); the underlying notion, linear mode connectivity, was developed to ask when training runs that differ only in SGD noise end in a linearly connected region (Frankle et al., 2020). Task arithmetic adds and subtracts the weight differences produced by fine-tuning on different tasks (Ilharco et al., 2023); TIES-merging and AdaMerging address the interference that appears when many such differences are combined (Yadav et al., 2023; Yang, E. et al., 2024). For rewards specifically, rewarded soups interpolate the weights of policies trained on different rewards (Rame et al., 2023), personalised soups merge parameters trained on separate preference dimensions (Jang et al., 2023), WARP merges rewarded policies at several stages of training (Rame et al., 2024b), and WARM averages reward models rather than policies (Rame et al., 2024a).

A third family trains one model to serve many weightings: MODPO folds the weighted objective into a DPO-style loss (Zhou, Z. et al., 2024b), Rewards-in-Context conditions on reward values in the prompt (Yang, R. et al., 2024), Panacea injects a preference vector through singular values of a low-rank adapter (Zhong et al., 2024), and conditional language policies train a steerable model by multi-task fine-tuning (Wang, K. et al., 2024). Safe RLHF treats harmlessness as a constraint rather than a weighted term (Dai et al., 2024). These methods choose the space of weightings before training. PoEM's distinguishing move is to fit the weights after training, from a new reward evaluated on a small calibration set, and to extend the claim to rewards that are not weighted sums of the basis.

3. Three Things "the RL Outcome" Can Mean

A forecast needs a target. Three are in play, and the published evidence speaks to them unequally.

T1, the optimum. The KL-regularised optimum for the new reward at a specified beta and reference. This is a mathematical object, defined whether or not anyone trains toward it. For a reward that is a linear combination of basis rewards, with every basis policy exactly optimal at a shared beta and reference, composition gives T1 exactly at the sequence level (Section 2.1).

T2, the run. The policy that a particular training recipe produces when run on the new reward from the reference. A recipe includes the algorithm, the form and strength of the KL term, reward normalisation, the number of steps, the adapter, and the seed. T2 is a random variable: two runs differing only in the order of training prompts produce different policies.

T3, the score. A summary of T2's behaviour: its mean reward under the target reward, its distance from the reference, or its rate of some behaviour.

Prediction of training outcomes also comes in more than one kind, and the kinds should not be confused. One extrapolates a run from its own beginning: Khatri et al. (2025) fit sigmoidal compute-performance curves to reinforcement-learning runs, report that stable recipes follow predictable trajectories, and use smaller runs to predict the validation performance of a single run scaled to 100,000 GPU-hours; Cai et al. (2025) extrapolate a run's final parameter update from an early window. A second predicts the result of training from the training data, as datamodels do with linear functions of which examples were included (Ilyas et al., 2022). A third, PoEM's kind, predicts a run on one reward from completed runs on other rewards, without any training on the new one. The first two forecast T2 or T3 from

evidence about the same run or the same learner; only the third forecasts a run that has not begun, from runs that optimised something else.

These targets come apart. A policy can match T3 without resembling T2, which PoEM recognises by reporting both reward recovery and per-token KL from the trained target (Hamidieh et al., 2026). T2 can differ from T1 because the recipe optimises a different objective or does not converge. And T2 has a spread: the gap between two runs bounds how precisely any method can forecast a single run.

Reward and distance are linked through the KL budget. Yang, J. Q. et al. (2024) show that any alignment method achieving a reward-KL trade-off comparable to the KL-constrained optimum must approximate that optimum in relative entropy. A composed policy that matches the target's reward at a similar KL from the reference is therefore close to T1; one that matches the reward at a larger KL need not be. Comparisons with best-of-n raise the same accounting issue, and the commonly quoted closed-form KL of the best-of-n policy is an upper bound rather than its actual value (Beirami et al., 2024). Recovery figures reported without the predictor's own KL from the reference leave the trade-off unstated.

PoEM's evaluation compares the composed policy against one trained run per target, so its main quantities measure T3 and, through relative KL, a single draw of T2. Its theory concerns T1. Its strongest numbers, on combined rewards, come with a seed baseline: a second training run with another seed has a median reward error of 0.12 to 0.17, against 0.19 to 0.28 for the composed policy on the programmatic bases (Hamidieh et al., 2026). That baseline is informative but narrow. On the programmatic GRPO basis the second seed changes only the order of training prompts, so it bounds the variance of T2 from below rather than describing it. On the nine combined rewards for which a second GRPO run exists, the composed policy with true weights matches that run's median reward error of 0.12, but the second run is closer to the first in distribution on six of the nine. On the DPO basis, the second run is only somewhat closer to the first than the reference model is (KL 0.54 against 0.69), and the authors note that distance to the target is a noisy quantity there. A forecast of T2 is being judged against an outcome that does not reproduce itself closely.

The distinction matters for the use PoEM proposes: previewing what a candidate reward would do before paying for training on it. A preview of T1 tells a practitioner what the idealised objective rewards. A preview of T2 tells them what their recipe will produce, including its failures. The evidence in Sections 4 to 6 bears on how far the first can stand in for the second.

4. Where Practice Leaves the Closed Form

The closed form assumes that each expert is the exact optimum of the same objective, a fixed reward minus beta times the sequence-level reverse KL to a shared reference, over an unrestricted policy class. PoEM's own discussion notes that finite capacity and imperfect optimisation break the last assumption, and that it therefore composes the implicit rewards the trained policies represent rather than the nominal rewards (Hamidieh et al., 2026). That move absorbs some departures and not others. Five are documented.

4.1 Reward scale

The tilt depends on r divided by beta. Doubling a reward's scale is the same as halving beta. Common recipes do not preserve scale. GRPO divides each reward by its standard deviation within the group of responses to a prompt (Shao, Z. et al., 2024), and DPO uses only which response in a pair is preferred (Rafailov et al., 2023). A policy trained with either receives updates of similar size whatever the reward's natural range, and a policy trained on a weighted combination of rewards receives updates of the same size as one trained on a single reward. The weighted average of the experts' log-ratios is not renormalised this way, so its magnitude shrinks as more weakly correlated experts are averaged.

PoEM documents this directly and corrects for it. At its default strength, recovery on the programmatic bases drops once more than two rewards are combined; a rescaling factor computed from the experts' Gram matrix, gamma_geom, restores recovery to 0.83 to 1.00 (Hamidieh et al., 2026). On the two reward-model bases with combined targets, which were trained with PPO and a KL penalty in the reward, keeping the reward's scale, the default strength already recovers 0.78 to 1.08 of the target's gain. The pattern fits the closed form: the bases whose recipe kept the quantity the closed form depends on needed no correction, and the bases whose recipe discarded it did. The authors show that gamma_geom is exact in one idealised case, rewards sharing a common within-prompt standard deviation; outside it, the factor is a heuristic, and one that grows with the number of composed experts, following the value expected for independent experts up to four and falling below it beyond, where the experts overlap. The reward-model bases differ from the programmatic ones in reward, prompts, response length and algorithm as well as in scale handling. Their own match to the closed form is also imperfect: the KL coefficient is adaptive, starting from a set value and tuned toward a KL target, so beta is neither fixed nor guaranteed to be shared across experts, and the RM-RB and RM-HH experts are initialised independently per run (Hamidieh et al., 2026). They contribute only eleven combined targets, so the comparison does not isolate scale. It is consistent with scale being one cause.

4.2 The KL term

The closed form needs the reverse KL from policy to reference at the sequence level. Recipes implement this term in several ways, and whether those implementations optimise the intended objective is disputed. Tang and Munos (2025) argue that differentiating through a KL estimate used as a loss, as several open-source implementations do, generally fails to produce the gradient of the KL, and that some implementations ignore the sequential structure of the estimate and obtain at best a partial gradient. Zhang, Y. et al. (2025) derive the weightings under which each estimator yields the exact gradient of a KL-regularised objective, show that the widely used k3 penalty corresponds to the unnormalised KL, and identify an importance-weighting mismatch in GRPO's KL term under off-policy updates. The two papers agree that implementation choices change which objective is optimised; they place the boundary of correctness differently. PoEM's programmatic GRPO basis uses the k3 estimator in the loss with coefficient 0.04, and its reward-model bases put a k3 estimate in the reward with an adaptive coefficient (Hamidieh et al., 2026). Under either reading, the GRPO experts' effective objective is not guaranteed to be the one whose optimum the composition assumes.

Many recipes drop the term. DAPO, an open large-scale recipe, removes the KL penalty (Yu et al., 2025); the diffusion experts in PoEM's image basis are trained with DDPO with no KL penalty (Hamidieh et al., 2026; Black et al., 2024). Without a KL term the closed form has no beta, and the optimum of pure reward maximisation is a degenerate distribution concentrated on the highest-reward responses, a collapse that motivated the KL term in the first place (Korbak et al., 2022). Trained policies do not reach that degenerate optimum, and one line of work explains why: on-policy reinforcement learning is implicitly biased toward solutions close in KL to the starting model, among the many that solve a task (Shenfeld et al., 2025). If that bias acts like an implicit KL penalty of roughly constant strength, composition could still work without an explicit term, but the strength would be set by the optimiser and the training length rather than by a coefficient, and nothing guarantees it is shared across experts trained on different rewards.

4.3 Offline training and implicit rewards

PoEM's DPO basis is trained offline, on fixed pairs of reference-model responses ranked by each reward. Its results differ from the GRPO basis in a consistent way: on combined rewards the composition matches the target's reward at gamma_geom but not its distribution (relative KL 1.05; closer than the reference on 13 of 32 targets), and on held-out rewards covered predictions reach the reward but not the distance (Hamidieh et al., 2026). The authors attribute the difference to training mode.

Independent work explains why offline-trained policies make weaker implicit rewards. Lin et al. (2024) compare DPO's implicit reward with an explicitly trained reward model and find that, while both fit the training preferences comparably, the implicit reward generalises worse under distribution shift, with a mean accuracy drop of 3% and a maximum of 7% across five out-of-distribution settings. Azar et al. (2024) argue that when preferences are deterministic or nearly so, the DPO objective can drive the policy away from the reference however beta is set, so the KL regularisation the closed form assumes is not enforced. Xu, S. et al. (2024) argue that DPO can find solutions that exploit responses outside the preference data's distribution, and report PPO outperforming DPO across their testbeds; Ivison et al. (2024) report PPO ahead of DPO by up to 2.5% in math and 1.2% in general domains. Rafailov et al. (2024) show that the choice of reference policy causes DPO's implicit rewards to decline during training. Each result describes a way in which beta times a DPO policy's log-ratio is a less faithful implicit reward than the closed form supposes, and PoEM's regression uses exactly that quantity as its features.

4.4 Token-level versus sequence-level normalisation

The composed optimum is defined over whole responses. Sampling from it exactly requires a normaliser over all continuations at every prefix. PoEM, like MOD, approximates it by applying the composition to next-token distributions and normalising at each step (Hamidieh et al., 2026; Shi et al., 2024). In the language of Section 2.2 this replaces a sum of soft Q-functions with a sum of per-token log-ratios, and the reinforcement-learning composition literature predicts an error that grows with the divergence between the component policies (Haarnoja et al., 2018; Hunt et al., 2019). The token-level reading of DPO makes the same point from the other side: in a token-level MDP, beta times the per-token log-ratio of an optimal policy plays the role of an optimal advantage rather than a reward (Rafailov et al., 2024), and optimal advantages for different rewards do not in general add the way the rewards do.

PoEM measures this gap directly, and the measurement is one of the most informative in the paper. Using importance weights to estimate the sequence-level product from decoder samples on three combined rewards, it finds that the two distributions differ by 0.6 to 2.7 nats per 96-token response, more for larger numbers of experts and larger gamma; that reweighting changes the combined reward by at most 0.03 standard units; and that the sequence-level product is further from the trained target than the token-level decoder in all six settings, significantly in five (Hamidieh et al., 2026). The authors conclude that local normalisation is not what limits the method on this basis.

The result has a second reading. If the exact T1 object is further from the trained run than its token-level approximation is, the trained run is not close to T1. One explanation, offered here as a conjecture, is that a policy trained by token-level policy gradient with a token-level KL estimate inherits the same local structure as the token-level decoder, so that the decoder approximates T2 better than T1 does. If so, the closed form is not the reason the composition works on this basis, and the success is evidence about how trained policies relate to each other rather than about the optimum. The measurement covers three targets, sixteen prompts and 96 tokens; whether the ordering holds for longer responses or for the reward-model bases is not reported.

4.5 The target run is chosen

Reward overoptimisation, the decline in true quality as a policy is pushed against an imperfect proxy reward, follows a different functional form for reinforcement learning than for best-of-n sampling, with coefficients that scale with the reward model's size (Gao, L. et al., 2023). KL regularisation is too weak to prevent it in offline alignment, on one theoretical analysis (Huang et al., 2025). Overoptimisation is a property of the run: it depends on how far and how long the policy is pushed.

PoEM's reward-model targets were selected to exclude it, and no hacking screen is reported for the programmatic targets. The full harmlessness run on one reward-model basis and the full SteamSHP run on another exploited their rewards, so the authors used checkpoints from step 100; candidates for the ten-member diverse basis had to pass a reward-hacking health check against four other judges and a test for surface tics before inclusion, and four retained mild tics: the SteamSHP and Starling experts add emoji sign-offs to 19 to 26% of benign responses, and the Skywork and creative-writing experts repeat an opening in 11 to 22% of responses on one half of the prompts (Hamidieh et al., 2026). These are sensible choices for testing whether composition reproduces well-behaved policies. They also mean that the forecast was never asked to predict the outcome a practitioner previewing an untested reward would most want to know about: whether training on it will hack it. A composition of experts that each passed a hacking check cannot exhibit a hack that none of them contains, and a coverage score fitted on samples from those experts has no way to register one.

5. Which Low Rank?

PoEM's headline geometric finding is a gap between spaces: the twenty programmatic experts' weight updates have effective rank 19.4 with a median pairwise cosine of 0.02, while their centred log-ratios have effective rank 6.3 (Hamidieh et al., 2026). Several published findings describe reinforcement-learning updates as low-rank, sparse or concentrated, and they are not the same finding.

Within one run, in parameter space. Mukherjee et al. (2025) find that reinforcement learning updates only 5 to 30 percent of parameters, across seven algorithms and ten models, that the subnetworks found under different seeds, data and algorithms overlap more than chance, and that the updates to almost every weight matrix are nonetheless nearly full-rank. Cai et al. (2025) find, across eight models and five reinforcement-learning algorithms, that the top singular direction of each matrix's update recovers over 99% of the reasoning gain and grows linearly during training, while the supervised fine-tuning and distillation updates they also examine need more directions; enough to extrapolate the final update from an early window with up to 2.5 times speed-up while keeping over 96% of the gain. Zhu et al. (2025) find that, for a fixed model, updates localise to consistent regions across runs and are largely invariant to datasets and recipes, avoiding the weights' principal directions. These findings concern the structure of a single update and its consistency across runs; they are compatible with each other because numerical rank and functionally dominant rank are different quantities.

Across rewards, in parameter space. PoEM's 19.4 of 20 says that experts trained on different rewards from one initialisation move in nearly orthogonal directions in the space of adapter weights (Hamidieh et al., 2026). This is a statement about LoRA updates of rank 64, trained for 28 steps; a low-rank adapter constrains each update but not the angle between updates.

Across rewards, in log-likelihood space. The 6.3 of 20 says that, evaluated on a fixed set of responses, the experts' changes to log-probability are highly correlated. This is the finding PoEM relies on.

The third finding is measured on a specific set of responses, and the choice matters. The geometry is computed on 2,400 responses sampled from the reference model for 600 prompts (Hamidieh et al., 2026). Appendix C.5 repeats the measurement on the experts' own responses for the two reward-model bases where it is reported. On the ten-member diverse basis, the log-ratio effective rank is 2.84 against 6.96 for the rewards on reference responses, but 6.09 against 8.00 on the experts' own responses. On the four-member RewardBench basis, the log-ratios have lower rank than the rewards on reference responses (1.64 against 2.30) and higher rank on the experts' own responses (2.77 against 2.43). Figure 1(a) shows these values. The rank gap that motivates extending composition beyond linear rewards narrows substantially on one basis and reverses on the other once the measurement moves to the responses the experts actually produce.

The reading matters because the composed policy is evaluated on its own samples, not on the reference model's. A subspace that is low-dimensional where the experts have not yet moved and higher-dimensional where they have is a subspace whose compression is

partly a property of the evaluation distribution. A plausible mechanism, offered as a conjecture, is that on reference responses every expert's log-ratio is dominated by a shared departure from the reference, so that one or two directions explain most of the variance, while on the experts' own responses the reward-specific differences that drove each expert become visible. PoEM's own description of its geometry supports the first half: sorting the log-ratio matrix by its first principal component shows nearly all experts moving together along one shared direction, and part of the shared structure is response length (Hamidieh et al., 2026). Coverage is a separate matter: PoEM computes it on a calibration set of responses sampled from the experts, not from the reference model. On the diverse basis it uses a "within-expert" variant, chosen before decoding, that first regresses the reward and the log-ratios on which expert wrote each response and then fits the ridge regression to what is left. On the same expert-sampled responses, this variant ranks recovery across the ten held-out rewards better than coverage centred per prompt (Spearman 0.81 against 0.45). Every within-expert value falls below the 0.3 threshold, so the basis is split at its median instead (Hamidieh et al., 2026).

6. The Shared-Drift Reading

6.1 What reinforcement learning does regardless of the reward

A growing set of results finds that much of what reinforcement learning does to a pretrained language model can be obtained without an informative reward.

Shao, R. et al. (2025) report that GRPO with randomly assigned rewards raises Qwen2.5-Math-7B's MATH-500 accuracy by 21.4 points, against 29.1 points with correct rewards, and trace the effect to a clipping bias that amplifies behaviours the model already favoured; in that case, reasoning written as code rose from 65% to over 90% of responses. They stress that the effect is model-dependent: rewards that work for Qwen models often fail for Llama and OLMo models. Agarwal et al. (2025) report that reinforcement learning with negative entropy as the only reward matches or exceeds GRPO and RLOO trained on 60,000 labelled examples on Qwen-7B. Zhao et al. (2025) report that using the model's own self-certainty as the sole reward in GRPO matches standard GRPO on mathematical benchmarks. Wang, Y. et al. (2025) report that training on a single example raises Qwen2.5-Math-1.5B from 36.0% to 73.6% on MATH500, the same as training on a 1,200-example subset containing it. Karan and Du (2025) report that an MCMC sampler drawing from a sharpened version of the base model's own likelihood, with no training, nearly matches and sometimes exceeds reinforcement learning on MATH500, HumanEval and GPQA.

Other work describes systematic, reward-independent changes that training induces. Policy entropy collapses early in training, and performance tracks an empirical function of entropy (Cui et al., 2025). An optimisation bias in GRPO inflates response length, especially for incorrect responses (Liu, Z. et al., 2025). Feedback training reduces output diversity relative to supervised fine-tuning (Kirk et al., 2024). Trained models outperform their bases at small sample counts while bases do better at large ones, consistent with training reweighting solutions the base already produces (Yue et al., 2025), and across several benchmarks more questions lose a verified solution than gain one during training (Wu, F. et al., 2025). Reinforcement learning through low-rank adapters on a 1.5-billion-parameter model is competitive with full-parameter reasoning models built on the same base, which Wang, S. et al. (2025) hypothesise reflects rapid adaptation to the structural format the reward favours. Reported gains are also sensitive to seeds, decoding parameters and evaluation details (Hochlehnert et al., 2025), a point made for deep reinforcement learning in general well before language models (Henderson et al., 2018).

Most of these results come from reasoning tasks with verifiable rewards, several come from one model family, and PoEM's rewards are stylistic programs and preference models. They establish that a reward-independent component of reinforcement learning exists and can be large under some conditions. They do not establish its size in PoEM's setting.

6.2 What a shared component would do to the evidence

Suppose each expert's log-ratio is the sum of a component d that every expert trained under a given recipe shares (format, length, reduced entropy, amplified high-prior behaviours) and a component s_k specific to reward k. Three predictions follow.

First, the log-ratio matrix would have a dominant direction, d, regardless of how the rewards relate, and its effective rank would be smallest where d dominates s_k most. That is what PoEM's first-component plot and its reference-versus-on-policy contrast show (Section 5).

Second, any composition with positive weights summing to one would reproduce d. The target run would contain d as well, because it was trained under the same recipe. A composed policy would therefore move toward the target, lowering its relative KL, whatever the weights, and would raise the target reward to the extent that d correlates with it.

Third, the improvement from fitting weights to the target reward, over composing without reference to it, would measure the reward-specific part.

PoEM reports the comparison that tests the third prediction on one basis. On the ten held-out reward models of the diverse basis, the composed policy with fitted weights recovers a median 0.56 of the held-out expert's reward gain and is closer to it than the single top

expert on nine of ten rewards; the top expert recovers 0.44; uniform weights, which never see the target reward, recover 0.50 and are as close in KL as the fitted composition; best-of-16 sampling, which queries the target reward, recovers 0.60 (Hamidieh et al., 2026). Figure 1(b) shows these values. On this basis the fitted weights add 0.06 of recovery over a composition that ignores the target, and no distance over it. On the four-member RewardBench basis, coverage is near zero for every held-out reward, yet the composition recovers 0.56 to 0.83 of the gain, which the authors attribute to the reward models largely agreeing (Hamidieh et al., 2026). On the programmatic GRPO combined rewards, the composition with fitted weights has a higher median reward error (0.28) than the expert with the largest true weight in the target (0.23), a baseline that itself uses knowledge of the target; only with the true weights does it do better (0.19).

PoEM's appendix adds a case that shows both readings at work. For two held-out members of the diverse basis, the other experts' log-ratios explain 82% and 92% of the held-out expert's own log-ratio, while the other rewards explain only 21% and 63% of the held-out reward, and the composed policy recovers 0.43 and 0.57 of the gain (Hamidieh et al., 2026). The authors present this as evidence that an expert can lie near the span of the others when its reward does not. It is also what a large shared component would produce: a log-ratio that is mostly d is well explained by other log-ratios that are mostly d, and recovering about half the reward gain is what d alone might supply.

None of this shows that fitted weights are useless. On the GRPO basis's held-out rewards, coverage strongly ranks reward error (Spearman -0.79) and covered predictions are closer to the target than the reference on nine of nine, and on combined rewards with known weights the composition beats the top expert on three of four bases (Hamidieh et al., 2026). The point is narrower. The published comparisons do not separate how much of the success comes from the reward-specific component, because the baselines that would separate it, uniform weights and composition from experts trained on uninformative rewards, were run on one basis or not at all.

6.3 The null basis

The control the shared-drift reading calls for is direct. Train a basis of experts under the same recipe on rewards that carry no information about any target: random rewards per response, as in Shao, R. et al. (2025), or each programmatic reward with its values permuted across responses within each prompt. Compute coverage, recovery and relative KL for the same targets using this null basis. Its coverage gives the floor of the score; its recovery and distance give the share of PoEM's results that a recipe's shared drift supplies. The reward-specific part of the forecast is the difference between the real basis and the null basis, not between the real basis and the reference model. For the GRPO basis, where clipping bias is the documented route to reward-independent change (Shao, R. et al., 2025), the null basis should use the same group normalisation and clipping as the real one.

7. Weight Space Versus Policy Space

PoEM motivates composing in log-probability space by the observation that averaging adapter weights degrades as more models are combined (Hamidieh et al., 2026). The weight-space literature supports the observation in part. MOD's theory shows that for most f-divergence regularisers, including reverse KL, the optimal policy for a weighted objective is not guaranteed to lie in the region spanned by interpolating the component policies' weights, and its experiments favour logit mixing over parameter merging (Shi et al., 2024). Interference among merged task vectors grows with their number (Yadav et al., 2023).

The same literature also reports weight interpolation working well under conditions that matter here. Rewarded soups find that policies fine-tuned from a shared pretrained initialisation on different rewards remain linearly connected, and that interpolating them traces a front close to that of retraining on each weighting, across text, image and control tasks with up to five rewards (Rame et al., 2023). Their appendix reports where it fails: the more similar the rewards, the larger the advantage over retraining, and with a reward that conflicts with the others, METEOR in captioning or an NSFW penalty in image generation, retraining does better than interpolation (Rame et al., 2023). Personalised soups merge parameters trained on separate preference dimensions (Jang et al., 2023), and WARM reports a 79.4% win rate for a policy trained against a weight-averaged reward model over one trained against a single reward model (Rame et al., 2024a).

Rewarded soups' boundary condition and PoEM's coverage score describe the same phenomenon in two spaces: composition succeeds when the target lies among what the components already do, and fails when the target asks for something they oppose or lack. PoEM states this for uncovered rewards: increasing the composition strength cannot add a direction the basis does not have, and even strength tuned on the target leaves uncovered rewards at recovery 0.60 and 0.39 on the two programmatic bases, against 1.04 and 0.88 for covered ones (Hamidieh et al., 2026). The difference between the spaces is in what the components are assumed to share. Weight interpolation needs a shared basin; log-ratio composition needs a shared reference and, for exactness, a shared objective. Neither guarantees that a composition reaches what training on the new reward would reach.

8. The Measured Values

Table 1 consolidates the published results the argument rests on. Each row is one result in its source's own metric. The last column is the author's classification of which part of the argument the row bears on: T3 or T2 (Section 3), D1 to D5 for the five departures (Section 4), R for rank (Section 5), S for shared drift (Section 6) and W for weight space (Section 7).

Three patterns stand out. The rows on the closed form's own terms, scale and KL form, line up with the bases' results: the reward-model bases, trained with the penalty in the reward and the scale kept, recover their targets at the default strength, and the bases trained with normalised or ranked rewards need a correction. The rank rows show the gap that motivates extending composition beyond linear rewards to be a property of where it is measured. And the held-out rows show recovery that is moderate when coverage is high, available in large part without a target-specific fit on the one basis where that was tested, and present on another basis where coverage is near zero.

The image row needs a caveat its source states: on the diffusion basis the weights are fitted at each denoising step to the held-out expert's own noise prediction, so its recovery measures how much of that expert lies in the span of the others, and is not a prediction made without the expert (Hamidieh et al., 2026). Span R2 exceeds 0.99 for every reward and does not track recovery (Spearman 0.12 across rewards). The composition also drops a correction term that the authors derive and show to be zero only in special cases. Du et al. (2023) showed that summing the scores of diffusion models does not in general give the score of the product distribution, that the sampler rather than the model is responsible for the resulting failures, and that Metropolis-corrected samplers repair many of them; composable diffusion had earlier composed models as energy functions (Liu, N. et al., 2022).

9. Reading a Composed-Policy Forecast

The analysis suggests a procedure for any claim that composing trained policies predicts the result of training on a new reward. Its purpose is to say which target the evidence reaches and how much of the success is reward-specific. Algorithm 1 states it.

```
Algorithm 1. Reading a composed-policy forecast
Input: basis B of experts; target reward r;
       trained target run(s) T; recipe R
step 1 OBJECTIVE - does R keep reward scale and use
       a sequence-level reverse KL at a shared beta?
       if not, record which (scale, KL form, KL
       absent, offline) and expect a correction
step 2 TARGET - is the claim about T1 (optimum),
       T2 (a run) or T3 (a score)? report each
       measured quantity against the right one
step 3 SEED BAND - train at least two runs of T
       differing in more than prompt order; a
       forecast closer than the band is at the floor
step 4 ON-POLICY RANK - compute rank and coverage on
       the experts' own and the target's responses,
       not only on reference responses
step 5 NULL BASIS - train experts under R on
       uninformative rewards; score coverage,
       recovery and distance for the same targets
step 6 FIT GAIN - compare fitted weights with
       uniform weights and the null basis; credit
       only the excess to reward-specific prediction
step 7 FAILURE MODES - include targets whose full
       runs hack or collapse; report whether the
       forecast shows the failure
Output: target reached, share attributable to the
        reward, conditions under which it holds
```

PoEM already runs parts of the procedure. Step 1 is partly addressed by its scale analysis and its gamma_geom correction. Step 2 is addressed by reporting reward recovery and relative KL separately. Step 3 is run on fifteen combined rewards with a second seed that changes prompt order on the GRPO basis. Step 4 is run for rank on two reward-model bases in an appendix, and coverage is already computed on responses sampled from the experts. Step 6 is run on one basis, with uniform weights. Steps 5 and 7 are not run; Step 7 is the opposite of the selection described in Section 4.5.

For the larger claim that a forecast can replace a run, the procedure adds one requirement that no step can satisfy on its own: the result must hold beyond 0.6-billion-parameter models, 96-token responses on the programmatic bases and greedy decoding on 30 held-out prompts, the settings in which PoEM's programmatic results were obtained (Hamidieh et al., 2026). The authors list scale and long-context generation among what they have not tested.

10. The Strongest Case Against This Paper

The most serious objection is that the shared-drift reading is not a criticism at all. A forecast that reproduces what training under a recipe does to every reward is still a forecast: if most of what reinforcement learning on a new reward produces is the recipe's common drift, then a method that captures that drift has predicted most of the outcome, and asking it to isolate the reward-specific part sets a bar the outcome itself does not meet. On this view the uniform-weights result vindicates composition rather than undercutting it.

There is force in this. Whether the common component counts as part of the prediction depends on what the forecast is for. If the purpose is to estimate what a run will look like, the common component is part of the answer. If the purpose is the one PoEM emphasises, previewing what a particular candidate reward would do so as to choose among rewards, then the common component is the same for every candidate and carries no information for the choice. The two purposes need different baselines: the reference model for the first, a null or uniform composition for the second. This paper argues that the published evidence is reported against the first baseline and is offered for the second purpose.

Table 1. Published values bearing on whether a composed policy forecasts a training run, one row per result, source named on each row.

Source	Setting	Quantity	Value	Bears on
Hamidieh et al., 2026	P-GRPO, 32 combined	reward error: fitted / true wts / top expert / BoN-16 / 2nd run	0.28 / 0.19 / 0.23 / 0.37 / 0.12 (9 targets)	T3, S
Hamidieh et al., 2026	P-DPO, 32 combined	reward error: fitted / true wts / top expert / 2nd run	0.19 / 0.21 / 0.32 / 0.17 (6 targets)	T3, D3
Hamidieh et al., 2026	P-GRPO vs P-DPO, combined	relative KL to target; closer than ref	0.24, 32/32 vs 1.05, 13/32	T2, D3
Hamidieh et al., 2026	RM-RB / RM-HH, combined	reward error, fitted weights	0.05 / 0.07	T3, D1
Hamidieh et al., 2026	RM bases, default strength	share of target gain recovered	0.78 to 1.08	D1
Hamidieh et al., 2026	programmatic, gamma_geom	share of target gain recovered	0.83 to 1.00	D1
Hamidieh et al., 2026	9 P-GRPO combined, 2nd seed	median error PoEM vs 2nd run; 2nd run closer	0.12 vs 0.12; 6 of 9	T2
Hamidieh et al., 2026	P-DPO, 2nd seed	KL to target: 2nd run vs reference	0.54 vs 0.69	T2
Hamidieh et al., 2026	P-GRPO, 3 targets	decoder vs sequence product, nats/response	0.6 to 2.7	D4
Hamidieh et al., 2026	same	sequence product further from target	6 of 6 (5 significant)	D4
Hamidieh et al., 2026	RM-HH, RM-Div	full runs exploited reward; used step	100	D5
Hamidieh et al., 2026	P-GRPO, 20 experts	effective rank: weight updates vs log-ratios	19.4 vs 6.3	R
Hamidieh et al., 2026	RM-Div, reference resp.	effective rank: log-ratios vs rewards	2.84 vs 6.96	R
Hamidieh et al., 2026	RM-Div, own resp.	effective rank: log-ratios vs rewards	6.09 vs 8.00	R
Hamidieh et al., 2026	RM-RB, ref. / own resp.	log-ratios vs rewards	1.64 vs 2.30 / 2.77 vs 2.43	R
Hamidieh et al., 2026	held-out, 3 bases	recovery covered vs uncovered	0.55-0.69 vs 0.22-0.43	T3
Hamidieh et al., 2026	held-out, 3 bases	Spearman, coverage vs reward error	-0.71 to -0.81	T3
Hamidieh et al., 2026	RM-Div held-out	recovery: top / uniform / fitted / BoN-16	0.44 / 0.50 / 0.56 / 0.60	S
Hamidieh et al., 2026	RM-RB held-out	coverage; recovery	below 0.12; 0.56 to 0.83	S
Hamidieh et al., 2026	SD-DDPO, 13 held out	mean recovery (range); span R2	0.67 (0.25-1.07); above 0.99	D2
Shao, R. et al., 2025	Qwen2.5-Math-7B, MATH-500	gain, random vs true reward	21.4 vs 29.1 points	S
Wang, Y. et al., 2025	Qwen2.5-Math-1.5B, MATH500	1 example vs 1.2k subset	73.6 vs 73.6% (from 36.0)	S
Mukherjee et al., 2025	7 algorithms, 10 models	share of parameters updated	5 to 30%	R
Cai et al., 2025	8 models, RL updates	gain recovered by rank-1 update	over 99%	R
Cai et al., 2025	AlphaRL extrapolation	speed-up; gain retained	up to 2.5x; over 96%	R
Shi et al., 2024	helpful assistant, 3 objectives	MOD over parameter merging	+12.8% (relative)	W
Liu, A. et al., 2024	Llama2-70B, 7B proxies	share of tuning gap closed	88%	W
Lin et al., 2024	5 OOD settings	DPO implicit vs explicit RM accuracy	mean -3%, max -7%	D3
Ivison et al., 2024	math / general	PPO over DPO	up to 2.5 / 1.2%	D3
Rame et al., 2024a	summarisation	win rate, WARM-trained vs single RM	79.4%	W

A second objection is that the departures in Section 4 are already handled. PoEM composes the implicit rewards the trained policies represent, not the nominal ones, so an expert's failure to reach its nominal optimum does not matter; its log-ratio defines the reward for which it is optimal (Hamidieh et al., 2026). This is correct for the sequence-level product, which is exactly optimal for the composed implicit reward, and PoEM proves a bound for it in terms of how well the target reward is approximated by that composed reward. It does not address the comparison that matters for a forecast: the target is not the optimum for the target reward but a trained run (T2), which departs from that optimum in the same ways the experts depart from theirs. Implicit-

reward composition guarantees closeness to T1 given good coverage. Closeness to T2 requires that the run's departures be predictable from the experts' departures, which is plausible when they share a recipe and is what Section 4.4's evidence hints at, but has not been shown.

A third objection concerns sources. The argument leans heavily on one paper's appendix, the reward-independent results come mostly from reasoning tasks and from Qwen models, and the rank contrast is reported for two bases with four and ten experts. These limits are real. The paper places its weight on the patterns that rest on more than one row: the alignment between scale handling and default-strength recovery across four language bases, the reversal of the rank gap on two bases, and the agreement between the uniform-weights baseline and the near-zero-coverage recovery as two observations of the same effect.

11. What Is Not Known

No published study has trained a null basis of experts on uninformative rewards and compared its coverage, recovery and distance with those of a real basis. The share of composition's success attributable to reward-specific structure is therefore unknown.

It is not known whether the log-ratio rank gap survives on on-policy responses for the programmatic bases, where the headline 6.3-of-20 figure was measured; the on-policy contrast is reported only for the two reward-model bases.

It is not known whether composition forecasts runs of models larger than 0.6 billion parameters, runs with longer responses, or runs without a KL term. PoEM names the first two as untested and notes that many runs omit the KL term. Its only basis trained without a KL term, the SD-DDPO image basis, fits its weights to the held-out expert itself, so it is not a forecast of a run trained without the term.

It is not known whether a composed policy can forecast reward hacking or collapse, because no published evaluation includes targets whose full runs exhibited them.

It is not known whether the sequence-level product's greater distance from the trained target generalises beyond three targets and 96 tokens, or what causes it.

It is not known how variable the target run is under seeds that change more than prompt order, so the floor against which a forecast should be judged has not been measured for these bases.

12. Studies That Would Settle It

Four studies, each within reach of PoEM's released setting, would replace the readings above with measurements.

First, the null basis of Section 6.3. Train twenty experts on the programmatic GRPO recipe with rewards permuted within each prompt, and twenty with random rewards, and run PoEM's combined and held-out evaluations with each basis in place of the real one. If the null bases' recovery and relative KL are close to the real basis's, the forecasts so far are mostly forecasts of shared drift. If they are near the reference model's, composition's success is reward-specific.

Second, the on-policy geometry for the programmatic bases. Recompute the effective ranks of rewards and log-ratios on responses sampled from each expert and from the composed policies, and recompute coverage on those responses. If the gap persists, the extension beyond linear rewards stands on firmer ground than the reward-model appendix suggests; if it closes, the effective rank measured on reference responses should be retired as evidence for that extension.

Third, a seed band with real variation. For a subset of combined rewards, train five target runs with different initial adapter seeds and data orders, and report the composition's error against the spread among them. A forecast that falls inside the band has predicted the run as well as the run predicts itself.

Fourth, failure targets. Include among the held-out targets the full, unselected runs that hacked their rewards, and report whether the composed policy, its coverage, or the experts' log-ratios give any sign of the hack before training. A forecasting method that cannot see the failure is still useful, but should be described as forecasting well-behaved runs.

13. Conclusion

The closed form of KL-regularised reinforcement learning makes a precise promise: if each policy is the optimum for its reward under a shared objective, the optimum for any linear combination of those rewards is their weighted product. Composition methods from DExperts to MOD use that promise, and PoEM extends it in two directions, by fitting the weights from a small calibration set and by observing that trained policies span few directions in log-likelihood space even when their rewards do not.

The promise is about an optimum. A practitioner who previews a reward wants to know about a run, and runs leave the closed form through scale normalisation, KL implementations, offline training, local normalisation and the failures that targets were chosen to exclude. PoEM's own results track those departures, which is evidence that the closed form explains part of why composition works and that the recipe explains another part. The low rank that licenses the move beyond linear rewards is measured mostly where the experts have not moved; where they have, it narrows on one basis and reverses on another. And on the one basis where a composition that never sees the target reward was

compared with one that does, the difference was six hundredths of the reward gain.

The evidence therefore supports a narrower claim than its title. Composing policies trained under one recipe reproduces much of what a further run under that recipe produces, and reproduces more when the target lies among what the experts already do. How much of that is a forecast of the reward and how much is a forecast of the recipe has not been measured. A null basis would measure it.

References

Agarwal, S., Zhang, Z., Yuan, L., Han, J., & Peng, H. (2025). The Unreasonable Effectiveness of Entropy Minimization in LLM Reasoning. arXiv:2505.15134. doi:10.48550/arXiv.2505.15134.

Azar, M. G., Rowland, M., Piot, B., Guo, D., Calandriello, D., Valko, M., & Munos, R. (2024). A General Theoretical Paradigm to Understand Learning from Human Preferences. arXiv:2310.12036. doi:10.48550/arXiv.2310.12036.

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., van Hasselt, H., & Silver, D. (2017). Successor Features for Transfer in Reinforcement Learning. arXiv:1606.05312. doi:10.48550/arXiv.1606.05312.

Beirami, A., Agarwal, A., Berant, J., D'Amour, A., Eisenstein, J., Nagpal, C., & Suresh, A. T. (2024). Theoretical guarantees on the best-of-n alignment policy. arXiv:2401.01879. doi:10.48550/arXiv.2401.01879.

Black, K., Janner, M., Du, Y., Kostrikov, I., & Levine, S. (2024). Training Diffusion Models with Reinforcement Learning. arXiv:2305.13301. doi:10.48550/arXiv.2305.13301.

Cai, Y., Cao, D., Xu, X., Yao, Z., Huang, Y., Tan, Z., Zhang, B., Sun, G., Liu, G., & Fang, J. (2025). On Predictability of Reinforcement Learning Dynamics for Large Language Models. arXiv:2510.00553. doi:10.48550/arXiv.2510.00553.

Chakraborty, S., Ghosal, S. S., Yin, M., Manocha, D., Wang, M., Bedi, A. S., & Huang, F. (2024). Transfer Q Star: Principled Decoding for LLM Alignment. arXiv:2405.20495. doi:10.48550/arXiv.2405.20495.

Christiano, P., Leike, J., Brown, T. B., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. arXiv:1706.03741. doi:10.48550/arXiv.1706.03741.

Cui, G., Zhang, Y., Chen, J., Yuan, L., Wang, Z., Zuo, Y., Li, H., Fan, Y., Chen, H., Chen, W., et al. (2025). The Entropy Mechanism of Reinforcement Learning for Reasoning Language Models. arXiv:2505.22617. doi:10.48550/arXiv.2505.22617.

Dai, J., Pan, X., Sun, R., Ji, J., Xu, X., Liu, M., Wang, Y., & Yang, Y. (2024). Safe RLHF: Safe Reinforcement Learning from Human Feedback. arXiv:2310.12773. doi:10.48550/arXiv.2310.12773.

Dekoninck, J., Fischer, M., Beurer-Kellner, L., & Vechev, M. (2024). Controlled Text Generation via Language Model Arithmetic. arXiv:2311.14479. doi:10.48550/arXiv.2311.14479.

Du, Y., Durkan, C., Strudel, R., Tenenbaum, J. B., Dieleman, S., Fergus, R., Sohl-Dickstein, J., Doucet, A., & Grathwohl, W. (2023). Reduce, Reuse, Recycle: Compositional Generation with Energy-Based Diffusion Models and MCMC. arXiv:2302.11552. doi:10.48550/arXiv.2302.11552.

Frankle, J., Dziugaite, G. K., Roy, D. M., & Carbin, M. (2020). Linear Mode Connectivity and the Lottery Ticket Hypothesis. arXiv:1912.05671. doi:10.48550/arXiv.1912.05671.

Gao, L., Schulman, J., & Hilton, J. (2023). Scaling Laws for Reward Model Overoptimization. arXiv:2210.10760. doi:10.48550/arXiv.2210.10760.

Gao, S., Ge, Q., Shen, W., Dou, S., Ye, J., Wang, X., Zheng, R., Zou, Y., Chen, Z., Yan, H., Zhang, Q., & Lin, D. (2024). Linear Alignment: A Closed-form Solution for Aligning Human Preferences without Tuning and Feedback. arXiv:2401.11458. doi:10.48550/arXiv.2401.11458.

Haarnoja, T., Pong, V., Zhou, A., Dalal, M., Abbeel, P., & Levine, S. (2018). Composable Deep Reinforcement Learning for Robotic Manipulation. arXiv:1803.06773. doi:10.48550/arXiv.1803.06773.

Hamidieh, K., Daras, G., & Torralba, A. (2026). PoEM: Predicting RL Outcomes from Existing Policies. arXiv:2609.30226. doi:10.48550/arXiv.2609.30226.

Han, S., Shenfeld, I., Srivastava, A., Kim, Y., & Agrawal, P. (2024). Value Augmented Sampling for Language Model Alignment and Personalization. arXiv:2405.06639. doi:10.48550/arXiv.2405.06639.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep Reinforcement Learning that Matters. arXiv:1709.06560. doi:10.48550/arXiv.1709.06560.

Hochlehnert, A., Bhatnagar, H., Udandarao, V., Albanie, S., Prabhu, A., & Bethge, M. (2025). A Sober Look at Progress in Language Model Reasoning: Pitfalls and Paths to Reproducibility. arXiv:2504.07086. doi:10.48550/arXiv.2504.07086.

Huang, A., Zhan, W., Xie, T., Lee, J. D., Sun, W., Krishnamurthy, A., & Foster, D. J. (2025). Correcting the Mythos of KL-Regularization: Direct Alignment without Overoptimization via Chi-Squared Preference Optimization. arXiv:2407.13399. doi:10.48550/arXiv.2407.13399.

Hunt, J. J., Barreto, A., Lillicrap, T. P., & Heess, N. (2019). Composing Entropic Policies using Divergence Correction. arXiv:1812.02216. doi:10.48550/arXiv.1812.02216.

Ilyas, A., Park, S. M., Engstrom, L., Leclerc, G., & Madry, A. (2022). Datamodels: Predicting Predictions from Training Data. arXiv:2202.00622. doi:10.48550/arXiv.2202.00622.

Ilharco, G., Ribeiro, M. T., Wortsman, M., Gururangan, S., Schmidt, L., Hajishirzi, H., & Farhadi, A. (2023). Editing Models with Task Arithmetic. arXiv:2212.04089. doi:10.48550/arXiv.2212.04089.

Ivison, H., Wang, Y., Liu, J., Wu, Z., Pyatkin, V., Lambert, N., Smith, N. A., Choi, Y., & Hajishirzi, H. (2024). Unpacking DPO and PPO: Disentangling Best Practices for Learning from Preference Feedback. arXiv:2406.09279. doi:10.48550/arXiv.2406.09279.

Jang, J., Kim, S., Lin, B. Y., Wang, Y., Hessel, J., Zettlemoyer, L., Hajishirzi, H., Choi, Y., & Ammanabrolu, P. (2023). Personalized Soups: Personalized Large Language Model Alignment via Post-hoc Parameter Merging. arXiv:2310.11564. doi:10.48550/arXiv.2310.11564.

Jaques, N., Ghandeharioun, A., Shen, J. H., Ferguson, C., Lapedriza, A., Jones, N., Gu, S., & Picard, R. (2019). Way Off-Policy Batch Deep Reinforcement Learning of Implicit Human Preferences in Dialog. arXiv:1907.00456. doi:10.48550/arXiv.1907.00456.

Karan, A., & Du, Y. (2025). Reasoning with Sampling: Your Base Model is Smarter Than You Think. arXiv:2510.14901. doi:10.48550/arXiv.2510.14901.

Khanov, M., Burapacheep, J., & Li, Y. (2024). ARGS: Alignment as Reward-Guided Search. arXiv:2402.01694. doi:10.48550/arXiv.2402.01694.

Khatri, D., Madaan, L., Tiwari, R., Bansal, R., Duvvuri, S. S., Zaheer, M., Dhillon, I. S., Brandfonbrener, D., & Agarwal, R. (2025). The Art of Scaling Reinforcement Learning Compute for LLMs. arXiv:2510.13786. doi:10.48550/arXiv.2510.13786.

Kirk, R., Mediratta, I., Nalmpantis, C., Luketina, J., Hambro, E., Grefenstette, E., & Raileanu, R. (2024). Understanding the Effects of RLHF on LLM Generalisation and Diversity. arXiv:2310.06452. doi:10.48550/arXiv.2310.06452.

Korbak, T., Perez, E., & Buckley, C. L. (2022). RL with KL penalties is better viewed as Bayesian inference. arXiv:2205.11275. doi:10.48550/arXiv.2205.11275.

Levine, S. (2018). Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review. arXiv:1805.00909. doi:10.48550/arXiv.1805.00909.

Li, X. L., Holtzman, A., Fried, D., Liang, P., Eisner, J., Hashimoto, T., Zettlemoyer, L., & Lewis, M. (2023). Contrastive Decoding: Open-ended Text Generation as Optimization. arXiv:2210.15097. doi:10.48550/arXiv.2210.15097.

Lin, Y., Seto, S., ter Hoeve, M., Metcalf, K., Theobald, B.-J., Wang, X., Zhang, Y., Huang, C., & Zhang, T. (2024). On the Limited Generalization Capability of the Implicit Reward Model Induced by Direct Preference Optimization. arXiv:2409.03650. doi:10.48550/arXiv.2409.03650.

Liu, A., Sap, M., Lu, X., Swayamdipta, S., Bhagavatula, C., Smith, N. A., & Choi, Y. (2021). DExperts: Decoding-Time Controlled Text Generation with Experts and Anti-Experts. arXiv:2105.03023. doi:10.48550/arXiv.2105.03023.

Liu, A., Han, X., Wang, Y., Tsvetkov, Y., Choi, Y., & Smith, N. A. (2024). Tuning Language Models by Proxy. arXiv:2401.08565. doi:10.48550/arXiv.2401.08565.

Liu, N., Li, S., Du, Y., Torralba, A., & Tenenbaum, J. B. (2022). Compositional Visual Generation with Composable Diffusion Models. arXiv:2206.01714. doi:10.48550/arXiv.2206.01714.

Liu, T., Guo, S., Bianco, L., Calandriello, D., Berthet, Q., Llinares, F., Hoffmann, J., Dixon, L., Valko, M., & Blondel, M. (2024). Decoding-time Realignment of Language Models. arXiv:2402.02992. doi:10.48550/arXiv.2402.02992.

Liu, Z., Chen, C., Li, W., Qi, P., Pang, T., Du, C., Lee, W. S., & Lin, M. (2025). Understanding R1-Zero-Like Training: A Critical Perspective. arXiv:2503.20783. doi:10.48550/arXiv.2503.20783.

Mitchell, E., Rafailov, R., Sharma, A., Finn, C., & Manning, C. D. (2024). An Emulator for Fine-Tuning Large Language Models using Small Language Models. arXiv:2310.12962. doi:10.48550/arXiv.2310.12962.

Mudgal, S., Lee, J., Ganapathy, H., Li, Y., Wang, T., Huang, Y., Chen, Z., Cheng, H.-T., Collins, M., Strohman, T., Chen, J., Beutel, A., & Beirami, A. (2024). Controlled Decoding from Language Models. arXiv:2310.17022. doi:10.48550/arXiv.2310.17022.

Mukherjee, S., Yuan, L., Hakkani-Tur, D., & Peng, H. (2025). Reinforcement Learning Finetunes Small Subnetworks in Large Language Models. arXiv:2505.11711. doi:10.48550/arXiv.2505.11711.

Nangue Tasse, G., James, S., & Rosman, B. (2020). A Boolean Task Algebra for Reinforcement Learning. arXiv:2001.01394. doi:10.48550/arXiv.2001.01394.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. arXiv:2203.02155. doi:10.48550/arXiv.2203.02155.

Peng, X. B., Kumar, A., Zhang, G., & Levine, S. (2019). Advantage-Weighted Regression: Simple and Scalable Off-Policy Reinforcement Learning. arXiv:1910.00177. doi:10.48550/arXiv.1910.00177.

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., & Finn, C. (2023). Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290. doi:10.48550/arXiv.2305.18290.

Rafailov, R., Hejna, J., Park, R., & Finn, C. (2024). From r to Q*: Your Language Model is Secretly a Q-Function. arXiv:2404.12358. doi:10.48550/arXiv.2404.12358.

Rame, A., Couairon, G., Shukor, M., Dancette, C., Gaya, J.-B., Soulier, L., & Cord, M. (2023). Rewarded soups: towards Pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. arXiv:2306.04488. doi:10.48550/arXiv.2306.04488.

Rame, A., Vieillard, N., Hussenot, L., Dadashi, R., Cideron, G., Bachem, O., & Ferret, J. (2024a). WARM: On the Benefits of Weight Averaged Reward Models. arXiv:2401.12187. doi:10.48550/arXiv.2401.12187.

Rame, A., Ferret, J., Vieillard, N., Dadashi, R., Hussenot, L., Cedoz, P.-L., Sessa, P. G., Girgin, S., Douillard, A., & Bachem, O. (2024b). WARP: On the Benefits of Weight Averaged Rewarded Policies. arXiv:2406.16768. doi:10.48550/arXiv.2406.16768.

Shao, R., Li, S. S., Xin, R., Geng, S., Wang, Y., Oh, S., Du, S. S., Lambert, N., Min, S., Krishna, R., Tsvetkov, Y., Hajishirzi, H., Koh, P. W., & Zettlemoyer, L. (2025). Spurious Rewards: Rethinking Training Signals in RLVR. arXiv:2506.10947. doi:10.48550/arXiv.2506.10947.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y. K., Wu, Y., & Guo, D. (2024). DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300. doi:10.48550/arXiv.2402.03300.

Shenfeld, I., Pari, J., & Agrawal, P. (2025). RL's Razor: Why Online Reinforcement Learning Forgets Less. arXiv:2509.04259. doi:10.48550/arXiv.2509.04259.

Shi, R., Chen, Y., Hu, Y., Liu, A., Hajishirzi, H., Smith, N. A., & Du, S. S. (2024). Decoding-Time Language Model Alignment with Multiple Objectives. arXiv:2406.18853. doi:10.48550/arXiv.2406.18853.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D. M., Lowe, R., Voss, C., Radford, A., Amodei, D., & Christiano, P. (2020). Learning to summarize from human feedback. arXiv:2009.01325. doi:10.48550/arXiv.2009.01325.

Tang, Y., & Munos, R. (2025). On a few pitfalls in KL divergence gradient estimation for RL. arXiv:2506.09477. doi:10.48550/arXiv.2506.09477.

Wang, K., Kidambi, R., Sullivan, R., Agarwal, A., Dann, C., Michi, A., Gelmi, M., Li, Y., Gupta, R., Dubey, A., et al. (2024). Conditional Language Policy: A General Framework for Steerable Multi-Objective Finetuning. arXiv:2407.15762. doi:10.48550/arXiv.2407.15762.

Wang, S., Asilis, J., Akgul, O. F., Bilgin, E. B., Liu, O., & Neiswanger, W. (2025). Tina: Tiny Reasoning Models via LoRA. arXiv:2504.15777. doi:10.48550/arXiv.2504.15777.

Wang, Y., Yang, Q., Zeng, Z., Ren, L., Liu, L., Peng, B., Cheng, H., He, X., Wang, K., Gao, J., Chen, W., Wang, S., Du, S. S., & Shen, Y. (2025). Reinforcement Learning for Reasoning in Large Language Models with One Training Example. arXiv:2504.20571. doi:10.48550/arXiv.2504.20571.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., & Schmidt, L. (2022). Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. arXiv:2203.05482. doi:10.48550/arXiv.2203.05482.

Wu, F., Xuan, W., Xing, D., Lu, X., Liu, M., Dong, Y., Harchaoui, Z., & Choi, Y. (2025). The Invisible Leash: Why RLVR May or May Not Escape Its Origin. arXiv:2507.14843. doi:10.48550/arXiv.2507.14843.

Xu, S., Fu, W., Gao, J., Ye, W., Liu, W., Mei, Z., Wang, G., Yu, C., & Wu, Y. (2024). Is DPO Superior to PPO for LLM Alignment? A Comprehensive Study. arXiv:2404.10719. doi:10.48550/arXiv.2404.10719.

Xu, Y., Sehwag, U. M., Koppel, A., Zhu, S., An, B., Huang, F., & Ganesh, S. (2025). GenARM: Reward Guided Generation with Autoregressive Reward Model for Test-time Alignment. arXiv:2410.08193. doi:10.48550/arXiv.2410.08193.

Yadav, P., Tam, D., Choshen, L., Raffel, C., & Bansal, M. (2023). TIES-Merging: Resolving Interference When Merging Models. arXiv:2306.01708. doi:10.48550/arXiv.2306.01708.

Yang, E., Wang, Z., Shen, L., Liu, S., Guo, G., Wang, X., & Tao, D. (2024). AdaMerging: Adaptive Model Merging for Multi-Task Learning. arXiv:2310.02575. doi:10.48550/arXiv.2310.02575.

Yang, J. Q., Salamatian, S., Sun, Z., Suresh, A. T., & Beirami, A. (2024). Asymptotics of Language Model Alignment. arXiv:2404.01730. doi:10.48550/arXiv.2404.01730.

Yang, R., Pan, X., Luo, F., Qiu, S., Zhong, H., Yu, D., & Chen, J. (2024). Rewards-in-Context: Multi-objective Alignment of Foundation Models with Dynamic Preference Adjustment. arXiv:2402.10207. doi:10.48550/arXiv.2402.10207.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L., et al. (2025). DAPO: An Open-Source LLM

Reinforcement Learning System at Scale. arXiv:2503.14476. doi:10.48550/arXiv.2503.14476.

Yue, Y., Chen, Z., Lu, R., Zhao, A., Wang, Z., Yue, Y., Song, S., & Huang, G. (2025). Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model? arXiv:2504.13837. doi:10.48550/arXiv.2504.13837.

Zhang, Y., Liu, Y., Yuan, H., Yuan, Y., Gu, Q., & Yao, A. C.-C. (2025). On the Design of KL-Regularized Policy Gradient Algorithms for LLM Reasoning. arXiv:2505.17508. doi:10.48550/arXiv.2505.17508.

Zhao, X., Kang, Z., Feng, A., Levine, S., & Song, D. (2025). Learning to Reason without External Rewards. arXiv:2505.19590. doi:10.48550/arXiv.2505.19590.

Zhu, H., Zhang, Z., Huang, H., Su, D., Liu, Z., Zhao, J., Fedorov, I., Pirsiavash, H., Sha, Z., Lee, J., et al. (2025). The Path Not Taken: RLVR Provably Learns Off the Principals. arXiv:2511.08567. doi:10.48550/arXiv.2511.08567.

Zheng, C., Wang, Z., Ji, H., Huang, M., & Peng, N. (2025). Model Extrapolation Expedites Alignment. arXiv:2404.16792. doi:10.48550/arXiv.2404.16792.

Zhong, Y., Ma, C., Zhang, X., Yang, Z., Chen, H., Zhang, Q., Qi, S., & Yang, Y. (2024). Panacea: Pareto Alignment via Preference Adaptation for LLMs. arXiv:2402.02030. doi:10.48550/arXiv.2402.02030.

Zhou, Z., Liu, Z., Liu, J., Dong, Z., Yang, C., & Qiao, Y. (2024a). Weak-to-Strong Search: Align Large Language Models via Searching over Small Language Models. arXiv:2405.19262. doi:10.48550/arXiv.2405.19262.

Zhou, Z., Liu, J., Shao, J., Yue, X., Yang, C., Ouyang, W., & Qiao, Y. (2024b). Beyond One-Preference-Fits-All Alignment: Multi-Objective Direct Preference Optimization. arXiv:2310.03708. doi:10.48550/arXiv.2310.03708.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., & Irving, G. (2019). Fine-Tuning Language Models from Human Preferences. arXiv:1909.08593. doi:10.48550/arXiv.1909.08593.

© 2026 Pranay Mahendrakar. Licensed under CC BY 4.0.

DOI: 10.5281/zenodo.24450652