ASAP: AMORTIZED DOUBLY-STOCHASTIC ATTENTION VIA SLICED DUAL PROJECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Doubly-stochastic attention has emerged as a transport-based alternative to row-softmax attention, with recent Transformer variants using it to reduce attention sinks and rank collapse while improving performance. In this family, the standard approach is Sinkhorn scaling, which trains more efficiently but still repeats matrix scaling in every inference forward pass. Sliced-transport attention removes the online iteration, but its soft sorting approximation materializes dense tensors for each slice, requiring substantially more training resources than Sinkhorn attention. We introduce *ASAP* (***A**mortized Doubly-**S**tochastic **A**ttention via Sliced Dual **P**rojection*), a train-then-compile method that trains the doubly-stochastic layer with Sinkhorn, then replaces the iterative scaling loop at inference with a fixed sliced-dual operator. It learns a lightweight parametric map from exact one-dimensional Kantorovich potentials to the Sinkhorn query-side dual, then reconstructs the attention plan with a two-sided entropic c-transform. Across language and vision benchmarks, ASAP keeps the cheaper training setup and remains highly competitive with recent baselines. In the main frozen-layer benchmark, ASAP is $5.3\times$ faster than the trained Sinkhorn teacher while matching its accuracy; in downstream replacements, ASAP recovers most of the teacher performance without any retraining.

1 INTRODUCTION

The Transformer is a standard architecture in modern language and vision systems (Vaswani et al., 2017; Dosovitskiy et al., 2021), and its success has made attention normalization a central design choice. Standard row-softmax attention normalizes each query independently, a one-sided constraint that recent work has associated with attention sinks in language models and rank collapse in deep attention stacks (Xiao et al., 2024; Dong et al., 2021; Lapenna et al., 2026). Doubly-stochastic attention adds the matching column constraint, providing a transport structure that has been shown to reduce these pathologies and improve performance across recent Transformer variants (Sander et al., 2022; Born et al., 2025). Trained row-softmax attention can also move toward this structure during optimization, suggesting that column balance is a useful inductive bias (Sander et al., 2022).

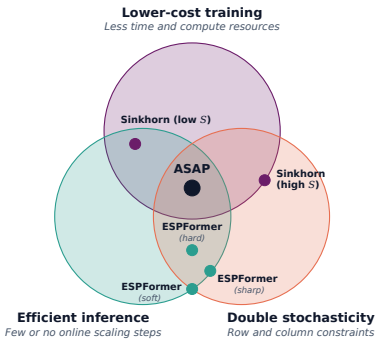


Figure 1: Attention tradeoffs.

The seminal Sinkformer work turns this bias into a trainable layer by replacing row-softmax normalization with a finite Sinkhorn scaling loop to obtain an entropic transport plan between query and key tokens (Sander et al., 2022; Cuturi, 2013). The Sinkhorn layer keeps the surrounding Transformer block unchanged, supports end-to-end training through Sinkhorn normalization, and has been shown to improve downstream performance across several benchmarks. However, a nonparametric Sinkhorn normalizer still repeats a finite scaling loop at inference, requiring $O(SN^2)$ scaling work for N tokens and S normalization steps on every call. Reducing S after training can lower latency, but it stops farther from the doubly-stochastic limit and changes the finite normalizer for the fixed projections. The practical question is how to avoid online scaling in a trained Sinkhorn layer while preserving its attention operator.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Other recent work has pursued non-iterative doubly-stochastic attention by changing the transport construction itself. ESPFormer introduces an elegant approach based on Expected Sliced Transport Plans, using sorted one-dimensional projections of queries and keys to form slice-wise matchings that are lifted and averaged across slices (Liu et al., 2025b; Shahbazi et al., 2025a). Its reported results show that sliced-transport attention can remain competitive across tasks, sometimes with improved performance, and its hard-sort inference operator is efficient, non-iterative, and exactly doubly stochastic. That benefit, however, comes after training with a soft sorting approximation, while inference switches to the hard doubly-stochastic matrix. Under the SoftSort operator used in ESPFormer, the training relaxation is row-stochastic at finite temperature and approaches the hard doubly-stochastic matrix only as the temperature is annealed. The soft-sort training operator materializes dense relaxation tensors for each slice, and in our matched IMDb resource profile against the $S = 10$ Sinkhorn teacher budget it uses $5.7\times$ more peak training memory and runs $14.8\times$ slower per training example than Sinkhorn attention. The larger training footprint significantly raises the cost of model selection, ablations, and repeated benchmark runs. This interesting line of work therefore leaves open a complementary question of whether non-iterative inference can be reached while keeping the cheaper training setup like Sinkhorn attention.

The two open questions above point to a doubly-stochastic attention trilemma. Current designs can have reasonable training cost, reduce online scaling, or stay close to a high-budget doubly-stochastic operator, but not all three at once. We introduce ASAP as a post-training answer to this trilemma. We instantiate this train-then-compile view with Sinkhorn teachers because they are the less expensive training setup in the current literature, and make the finite dual variables available for compilation. Given a trained, high-performing Sinkhorn layer, we freeze it and run a small unlabeled calibration set through the layer. From each forward pass, we extract the query-side normalization vector that the Sinkhorn loop would otherwise recompute; in optimal-transport terms, this is the source dual. On the same query and key activations, we compute exact one-dimensional transport potentials from sorted projections and use them as sliced geometric features. We then learn a lightweight parametric map from these features to the extracted Sinkhorn vector. At inference, the compiled layer predicts this one vector and applies a two-sided entropic c-transform under the teacher's finite Sinkhorn convention. This keeps the cheaper Sinkhorn training setup, replaces most of the online scaling loop, and preserves the matched marginal by construction.

In summary, our specific contributions are as follows.

- We introduce ASAP, a train-then-compile method for doubly-stochastic attention that keeps the Sinkhorn training setup and replaces the iterative online scaling loop after training while staying close to the trained Sinkhorn operator.
- We instantiate sliced-dual projection for frozen Sinkhorn attention, regressing exact one-dimensional Kantorovich potentials onto the teacher's *finite-budget* source dual and recovering the key dual analytically through the entropic c-transform.
- One frozen teacher supports multiple ASAP variants without retraining. Least-squares or convex-KL can be used to calibrate the sliced-dual map, and a one-sided or two-sided c-transform can be selected at inference.
- We demonstrate ASAP across multiple language and vision benchmarks, showing significant layer speedups over the trained Sinkhorn teacher with matched-resource comparisons against iterative Sinkhorn and sliced-transport baselines.

2 Related Work

Transport-based attention. Sinkformers apply Sinkhorn scaling to query–key scores (Sander et al., 2022); Sparse Sinkhorn Attention instead uses balancing for block-wise routing (Tay et al., 2020). ESPFormer lifts and averages sorted one-dimensional matchings (Shahbazi et al., 2025a), and LOTFormer uses a low-rank pivot coupling (Shahbazi et al., 2025b). These approaches change the attention construction. ASAP keeps the trained Sinkhorn projections and approximates their finite normalizer. FlashSinkhorn accelerates entropic OT through fused kernels (Ye et al., 2026); we include it as an implementation control rather than assuming that fewer normalization steps imply the fastest kernel.

Amortized and sliced transport. Meta OT predicts OT potentials or maps from previous transport problems (Amos et al., 2023), while learned and data-dependent initializers reduce the number of Sinkhorn iterations needed for new problems (Thornton & Cuturi, 2023). Sliced quantities have also been used as features for Wasserstein embeddings and distance prediction (Haviv et al., 2024; Nguyen et al., 2026). Our feature construction follows Truong & Nguyen (2026), who predict Kantorovich potentials from sliced potentials. We apply this idea to a trained attention layer, learning the potentials produced by its usual Sinkhorn computation while retaining its score scale and padding mask. Appendix C discusses the broader sliced-transport and attention literature.

3 Sinkhorn Attention

For an unmasked attention head, let $Q, K \in \mathbb{R}^{N\times d_h}$ and $V \in \mathbb{R}^{N\times d_v}$ have rows q_i, k_j, v_j. Standard attention applies softmax to each row of the score matrix $s_{ij} = q_i^\top k_j/\sqrt{d_h}$, allowing each query to distribute one unit of attention across the keys. The column sums are unconstrained, however, so some keys may receive much more attention than others. Sinkhorn attention addresses this imbalance by alternately normalizing the rows and columns of $\exp(s_{ij}/\varepsilon)$ toward unit sums (Sander et al., 2022). We can also interpret the normalized matrix as a transport plan: after rescaling A to $P = A/N$, each query and key has target mass $1/N$, giving marginals $\mu = \nu = \mathbf{1}_N/N$. The normalization must be repeated for every new input because the queries and keys, and hence the matrix to be balanced, change with that input. Our goal is to learn part of this computation after training. We take the trained model's usual T-step output $A^{(T)}$ as the reference, retain its score scale and update sequence, and use $\varepsilon = 1$ unless stated otherwise.

To identify what we need to learn, we use the matrix-scaling representation of entropic transport (Cuturi, 2013). The row and column rescalings are described by a *query potential* f and a *key potential* g, the source and target dual potentials in OT terminology. With the quadratic cost $C_{ij} = \|q_i - k_j\|_2^2/(2\sqrt{d_h})$, the plan has entries $P_C(f,g)_{ij} = \exp((-C_{ij} + f_i + g_j)/\varepsilon)$, where $e^{f_i/\varepsilon}$ scales row i and $e^{g_j/\varepsilon}$ scales column j. This cost differs from $-s_{ij}$ by query-dependent and key-dependent terms that can be absorbed into the potentials; Appendix F describes the coordinate conversion we use for the trained model. Although the plan depends on both vectors, we do not need to predict them independently. For any query potential f, we can determine g from the required column sums:

$$T_K^\nu(f)_j = \varepsilon \log \nu_j - \varepsilon \log \sum_{i=1}^{N} \exp\left(\frac{-C_{ij} + f_i}{\varepsilon}\right). \tag{1}$$

Here T_K^ν is the entropic c-transform, or column normalization expressed in potential coordinates. A teacher whose last step normalizes columns therefore satisfies $g^{(T)} = T_K^\nu(f^{(T)})$ and $A^{(T)} = NP_C(f^{(T)}, g^{(T)})$. We use this relation to reduce the learning problem from a full attention matrix to one potential vector: the predictor supplies f, and the column normalization supplies g. The remaining question is how to predict f without repeating the Sinkhorn iterations, which motivates the sliced-potential features introduced next.

4 Amortized Sinkhorn Attention

ASAP learns to predict the query potentials from simpler transport problems obtained by projecting queries and keys onto one-dimensional lines. Their potentials can be computed efficiently by sorting, and together they describe the geometry that the fitted predictor uses to approximate the trained layer. Figure 2 illustrates how this prediction replaces most of the repeated normalization at inference.

4.1 Sliced-Potential Features

For a fixed set of directions $\Theta = \{\theta_\ell\}_{\ell=1}^L$, with $\theta_\ell \in \mathbb{S}^{d_h-1}$, the projected queries and keys are

$$a_i^{(\ell)} = \frac{\theta_\ell^\top q_i}{d_h^{1/4}}, \qquad b_j^{(\ell)} = \frac{\theta_\ell^\top k_j}{d_h^{1/4}}. \tag{2}$$

We scale the projections to align their one-dimensional quadratic cost with C, then sort them to obtain $a_{(r)}^{(\ell)}$ and $b_{(r)}^{(\ell)}$. For uniform one-dimensional OT, we pair points of equal rank and compute the

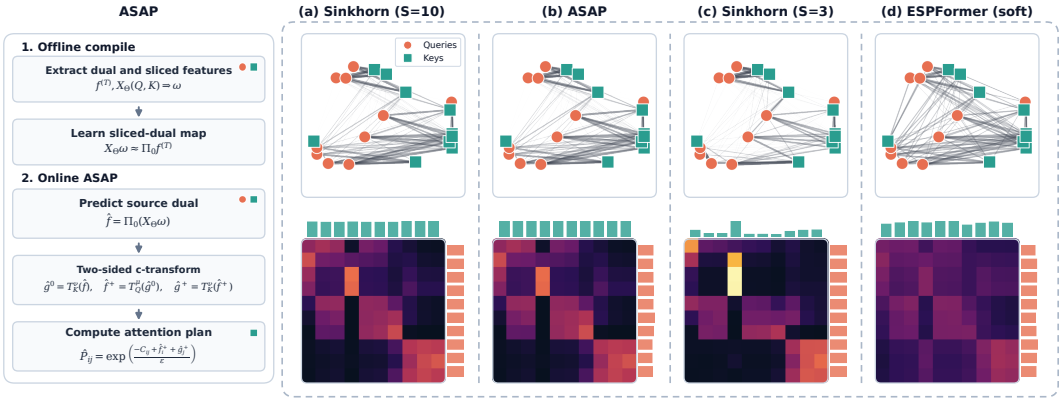


Figure 2: ASAP learns to predict a trained Sinkhorn layer's query potentials from one-dimensional transport potentials. At inference, it combines this prediction with a fixed number of normalization steps to reconstruct attention. The right panels show attention and marginal profiles for one query–key example; the experiments evaluate agreement with the teacher on held-out activations.

associated query potential as

$$\begin{aligned} \phi_1^{(\ell)} &= 0, \qquad \phi_r^{(\ell)} = \sum_{t=1}^{r-1} b_{(t)}^{(\ell)} \big(a_{(t+1)}^{(\ell)} - a_{(t)}^{(\ell)}\big), \\ f_{(r)}^{(\ell)} &= \tfrac{1}{2}\big(a_{(r)}^{(\ell)}\big)^2 - \phi_r^{(\ell)}, \qquad r = 1, \dots, N. \end{aligned} \tag{3}$$

We put these values back in the original query order and subtract their mean to obtain $\tilde{f}^{(\ell)}$, then stack the potentials across directions into $X_\Theta(Q,K) = [\tilde{f}^{(1)} \ \cdots \ \tilde{f}^{(L)}] \in \mathbb{R}^{N\times L}$. The predictor uses these features to estimate the teacher's query potential before we reconstruct attention. For each input, we spend $O(LNd_h + LN\log N)$ on the projections and sorting.

4.2 Learning the Query Potentials

The sliced potentials provide inexpensive descriptions of the query–key geometry, but we still need to learn how they combine to reproduce the potential used by the trained layer. We formulate this as a regression problem: for M unlabeled calibration inputs, we run the frozen Sinkhorn model and record its query potentials $f^{(T)}$, together with the feature matrices $X_\Theta(Q,K)$ computed from the same queries and keys. Each token supplies a row of features and a target value, while a shared coefficient vector $\omega \in \mathbb{R}^L$ determines how the slices contribute across inputs. Before fitting, we remove the arbitrary additive constant in the potentials. A constant shift in f is offset by the opposite shift in g under Equation 1, so we can center the targets and predictions without altering the reconstructed plan. With $\Pi_0(z) = z - \mathbf{1}_N(\mathbf{1}_N^\top z)/N$, we use

$$\tilde{f}^{(T)} = \Pi_0(f^{(T)}), \qquad \hat{f}_\omega(Q,K) = \Pi_0\big(X_\Theta(Q,K)\omega\big). \tag{4}$$

We pool token instances to fit the coefficients, either sharing a predictor across the heads of a layer or fitting each head separately. Both choices use the same regression, with L coefficients per predictor. The original model weights remain frozen throughout this procedure, and the calibration inputs require no task labels.

We consider two ways to fit this predictor, both using the same sliced features and the same frozen teacher. Our default, ASAP-LS, directly approximates the teacher's query potential through ridge regression with regularization parameter $\lambda > 0$. We collect the feature rows and centered teacher values into $\bar{X}$ and $\bar{y}$, so the fit reduces to a linear system involving only the L coefficients, regardless of the number of token-level examples collected. We use this approach as the default because we can

obtain the coefficients without iterative optimization. The objective and its solution are

$$\omega^{\mathrm{LS}} \in \arg\min_{\omega} \sum_{m=1}^{M} \left\| X_\Theta(Q^{(m)}, K^{(m)})\omega - \tilde{f}^{(T,m)} \right\|_2^2 + \lambda\|\omega\|_2^2. \tag{5}$$

$$\omega^{\mathrm{LS}} = (\bar{X}^\top \bar{X} + \lambda I_L)^{-1} \bar{X}^\top \bar{y}. \tag{6}$$

As an alternative, ASAP-KL fits the coefficients through the attention plan: we predict the query potential, recover the key potential with one column normalization, and minimize the KL divergence from the teacher's plan to this reconstruction. The KL fitting objective is convex in ω (Appendix B); it accounts for how potential errors affect the plan, whereas LS measures those errors directly in potential coordinates. Once fitted, either predictor can be used with the inference variants below.

4.3 Online Reconstruction

At inference, the fitted map supplies $\hat{f}_\omega$ and the key potential follows from the column normalization in Equation 1. The two potentials determine a plan through

$$P_C(f, g)_{ij} = \exp\left(\frac{-C_{ij} + f_i + g_j}{\varepsilon}\right), \tag{7}$$

and the corresponding row-normalization step is

$$T_Q^\mu(g)_i = \varepsilon \log \mu_i - \varepsilon \log \sum_j \exp\left(\frac{-C_{ij} + g_j}{\varepsilon}\right). \tag{8}$$

The column transform determines the key potential from our prediction, but the row sums still depend on the quality of that prediction. We therefore use the query transform to normalize the rows, then recompute the key potential for the updated query potential. This two-sided reconstruction has a fixed sequence of three transforms:

$$\hat{g}_\omega^0 = T_K^\nu(\hat{f}_\omega), \qquad \hat{f}_\omega^+ = T_Q^\mu(\hat{g}_\omega^0), \qquad \hat{g}_\omega^+ = T_K^\nu(\hat{f}_\omega^+). \tag{9}$$

The resulting potentials give the attention weights and head output,

$$\hat{P}_\omega = P_C(\hat{f}_\omega^+, \hat{g}_\omega^+), \qquad \hat{A}_\omega = N\hat{P}_\omega, \qquad \hat{Y} = \hat{A}_\omega V. \tag{10}$$

For a teacher whose last step normalizes rows, we instead return $P_C(\hat{f}_\omega^+, \hat{g}_\omega^0)$ after the query transform. We also evaluate a one-sided ablation, ASAP-0, which uses the same fitted predictor but only the initial column transform:

$$\hat{P}_\omega^0 = P_C(\hat{f}_\omega, T_K^\nu(\hat{f}_\omega)). \tag{11}$$

For a column-ending teacher, ASAP-0 recovers its attention matrix when the predicted query potential differs from the teacher's only by a constant. The two-sided construction also normalizes over queries, so it need not reproduce the teacher's finite-step output even under perfect prediction. Our experiments compare both reconstructions using the same fitted potentials.

We relate prediction error to attention error by comparing the plans $P_f = P_C(f, T_K^\nu(f))$ and $P_{\hat{f}}$. For unmasked inputs with positive marginals and $\|\hat{f} - f\|_\infty \le \alpha$, we obtain

$$e^{-2\alpha/\varepsilon} P_f \;\le\; P_{\hat{f}} \;\le\; e^{2\alpha/\varepsilon} P_f \quad \text{entrywise.} \tag{12}$$

Thus a small potential error produces a small multiplicative change in the plan, with sensitivity set by ε. With fixed slices and a consistent treatment of ties, the unmasked construction is also permutation equivariant. Appendix B proves these properties, bounds the error of the two-sided reconstruction, and states the approximation assumptions used in its random-feature analysis.

After score formation, Sinkhorn spends $O(SN^2)$ work on normalization, while ASAP needs $O(LNd_h + LN\log N + NL + RN^2)$ for its features, prediction, and transforms. Default ASAP uses $R = 3$ passes for a column-ending teacher and $R = 2$ for a row-ending teacher; ASAP-0 uses $R = 1$. In the dense implementation, we store the score matrix and compute each transform directly. In the streamed implementation, we evaluate the same transforms in tiles and accumulate the value-weighted output without storing the full matrix. Both retain quadratic arithmetic for query–key interactions; streaming reduces intermediate storage. The predictor and its offline fit are the same in either implementation (Appendix E.3).

5 EXPERIMENTS

We evaluate ASAP for repeated inference from an existing Sinkhorn checkpoint. We fit the potential predictor on unlabeled training inputs with the model's weights fixed, then measure attention error, classification accuracy, and inference cost. Unless stated otherwise, we use 32 slices and 4096 calibration inputs. The text and smaller vision benchmarks report mean and standard deviation over three runs; the ImageNet studies report individual runs. Training and evaluation settings are given in Appendices G and I.

5.1 ATTENTION APPROXIMATION

We begin with the attention layers of three trained IMDb encoders, using $S = 20$ Sinkhorn updates and sequence length 512. We record 144 held-out batch–layer cases and evaluate every replacement on the same query, key, and value activations, with the output projection unchanged. Table 1 compares ASAP with the teacher and with shorter Sinkhorn runs on these weights. Output RMSE measures the change in the value-weighted representation, while relative ℓ_2 error measures the change in the attention matrix.

Table 1: Attention replacement on IMDb at sequence length 512. All rows reuse the $S = 20$ teacher's projections. Latency is per layer-batch; output and attention errors summarize 144 held-out batch–layer cases, and accuracy summarizes three trained teachers. Entries are mean $\pm$ standard deviation.

Method	Time (ms) $\downarrow$	Output RMSE $\downarrow$	Attention rel. ℓ_2 $\downarrow$	IMDb acc. (%) $\uparrow$
Sinkhorn teacher ($S = 20$)	63.02 ± 0.12	–	–	87.34 ± 0.27
ASAP-LS ($L = 32$)	11.79 ± 0.02	0.003 ± 0.004	0.035 ± 0.070	87.36 ± 0.21
Sinkhorn ($S = 3$)	14.14 ± 0.03	0.068 ± 0.030	1.527 ± 0.319	86.92 ± 0.12
Sinkhorn ($S = 5$)	19.87 ± 0.05	0.060 ± 0.015	1.430 ± 0.241	87.15 ± 0.14
ASAP-KL ($L = 32$)	11.79 ± 0.02	0.003 ± 0.005	0.039 ± 0.072	87.34 ± 0.24
ASAP-LS-0 ($L = 32$)	7.74 ± 0.02	0.020 ± 0.017	0.364 ± 0.255	87.29 ± 0.20
ASAP-LS-0 ($L = 64$)	9.19 ± 0.02	0.018 ± 0.012	0.322 ± 0.177	87.25 ± 0.20
ASAP-KL-0 ($L = 32$)	7.74 ± 0.02	0.020 ± 0.016	0.365 ± 0.239	87.29 ± 0.24

Default ASAP-LS reduces layer-batch latency from 63.02 to 11.79 ms, a $5.3\times$ speedup, with output RMSE 0.003. IMDb accuracy is $87.36 \pm 0.21\%$, compared with the teacher's $87.34 \pm 0.27\%$. Shorter Sinkhorn runs give a less favorable compromise: three updates take 14.14 ms with RMSE 0.068, and five take 19.87 ms with RMSE 0.060. The one-sided ASAP-LS-0 is faster still at 7.74 ms, but its RMSE rises to 0.020 with the same potential predictor. The LS and KL fits give similar results, supporting the simpler LS calibration used by default. Table H.1 reports marginal errors under the teacher's padding rule.

5.2 CLASSIFICATION PERFORMANCE

We next replace attention throughout the trained models and evaluate their classification accuracy. On ImageNet-1k (Russakovsky et al., 2015), we use a DeiT-S/16 (Touvron et al., 2021) fine-tuned with Sinkhorn attention for five epochs. Default ASAP-LS reaches 77.48% top-1 accuracy without final row normalization, compared with 77.51% for the teacher on the same 25,000 validation images (Table 2). In a separate resolution study, we refit at each input size and find that ASAP stays within one percentage point of the teacher over 197–785 tokens (Appendix I).

We also evaluate a variant that normalizes the reconstructed attention rows before value mixing. This variant reaches 77.56% on 25,000 ImageNet-1k validation images. On ImageNet-100, a teacher trained from scratch for 30 epochs reaches 61.38%, compared with 61.28% after replacement. In a

Table 2: ImageNet top-1 accuracy (%) from individual runs. Each row compares the teacher and its replacement on the same validation images. Final row normalization applies only to ASAP, before value mixing.

Dataset	Final row norm.	Images	Teacher ($S = 10$)	ASAP-LS
ImageNet-1k	No (default)	25,000	77.51	77.48
ImageNet-1k	Yes	25,000	77.51	77.56
ImageNet-100	Yes	5000	61.38	61.28

separate ImageNet-100 run, we train the backbone and ASAP coefficients jointly and obtain 64.40% accuracy (Appendix I).

Text classification. For DBpedia-14, AG News, and Yelp Review Full, we train six-layer encoders at sequence length 128 and replace the attention of the $S = 10$ teachers after training. In Table 3, default ASAP-LS stays within 0.1 percentage points of the teacher on all three datasets, whereas the one-sided ASAP-LS-0 leaves gaps of 0.45–0.67 points with the same fitted predictor. The LS and KL fits are close for either reconstruction. Independently trained Transformer, DiffTransformer, and ESPFormer models provide task-level comparisons under the same training recipe.

Table 3: Classification accuracy (mean $\pm$ standard deviation, %) over three runs with task-specific training. Text encoders use sequence length 128; CIFAR-100 uses patch size 4. ASAP calibration uses 4096 unlabeled examples.

	Text classification			Vision
Method	DBpedia-14	AG News	Yelp Review Full	CIFAR-100
Sinkhorn teacher ($S = 10$)	98.27 ± 0.02	90.12 ± 0.12	54.54 ± 0.25	64.95 ± 0.06
ASAP-LS	98.26 ± 0.03	90.03 ± 0.17	54.51 ± 0.35	63.46 ± 0.36
Transformer	96.99 ± 0.42	89.34 ± 0.20	54.40 ± 0.05	62.73 ± 0.01
DiffTransformer	96.80 ± 0.33	88.01 ± 0.15	54.30 ± 0.06	60.03 ± 0.59
ESPFormer	96.58 ± 0.01	86.50 ± 0.43	49.49 ± 0.38	58.46 ± 0.42
ASAP-KL	98.25 ± 0.04	90.04 ± 0.18	54.46 ± 0.37	63.34 ± 0.44
ASAP-LS-0	97.82 ± 0.05	89.61 ± 0.18	53.87 ± 0.47	63.64 ± 0.42
ASAP-KL-0	97.90 ± 0.06	89.59 ± 0.20	53.83 ± 0.44	63.58 ± 0.66

Other vision benchmarks. The smaller vision tasks show a wider range of accuracy changes. On Cats and Dogs, two-sided ASAP with 64 slices gives a $1.53\times$ full-model speedup with a 0.65-percentage-point accuracy loss relative to its teacher (Table 4). On CIFAR-100 (Table 3), the teacher reaches 64.95%, compared with 63.46% for default ASAP-LS and 63.64% for ASAP-LS-0. Both reconstructions leave an accuracy gap here, unlike the text benchmarks. Appendix G.1 gives the full vision comparisons, and Appendix E.7 reports the effect of patch size on MNIST.

5.3 Training and Inference Cost

ASAP uses the Sinkhorn training run and fits its predictor afterward, so the training cost is that of the chosen teacher. Figure 3a compares full-encoder IMDb training at sequence length 512 and effective batch size 32. Sinkformer with $S = 10$ uses 12.24 GB and 9.37 ms per training example, compared with 69.51 GB and 138.82 ms for ESPFormer's soft-sort training path. ESPFormer retains a dense sorting matrix for each slice during training, whereas Sinkhorn scales one attention matrix per head. ESPFormer's hard-sort inference path is evaluated separately in Figure E.1; Appendix E.5 gives the full training measurements.

We then measure inference with the same FlashSinkhorn kernels for both the iterative teacher and ASAP. Figure 3b shows that default ASAP is $1.40\times$ faster than fused $S = 10$ at 4096 tokens and $2.05\times$ faster at 65536 tokens; ASAP-0 reaches $3.02\times$ at the latter length. Default ASAP uses 0.246 GB at 65536 tokens. Both methods avoid storing the dense attention matrix through streaming, while ASAP reduces the number of normalization passes. Relative to $S = 3$, default ASAP has the same pass count. Appendix E.3 gives the measurement protocol, memory results, and comparisons with dense PyTorch.

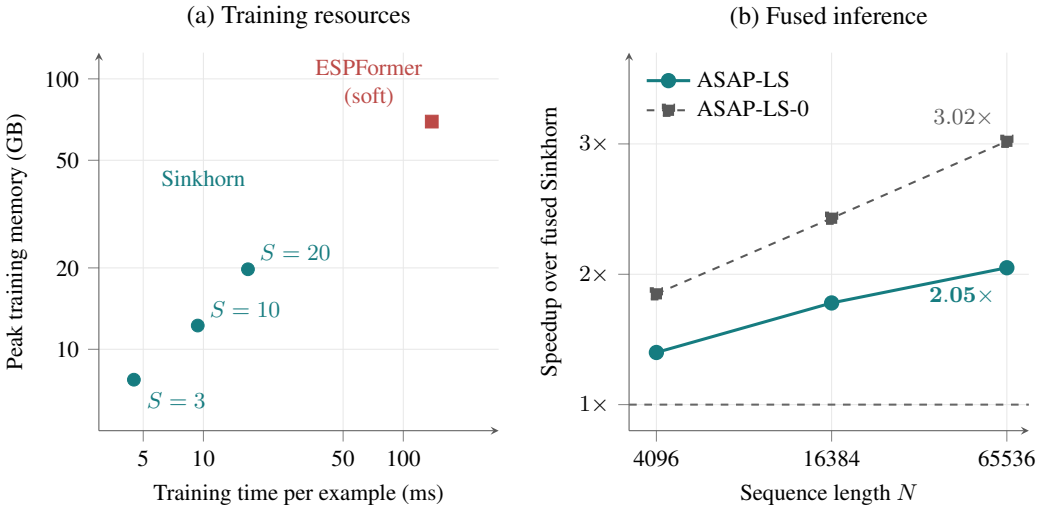


Figure 3: Training resources and inference speed. (a) Full-encoder IMDb training at length 512 and effective batch size 32 on RTX PRO 6000 Blackwell; ASAP uses its teacher's training run. Both axes are logarithmic. (b) Speedup over $S = 10$ Sinkhorn using the same FlashSinkhorn kernels on RTX 6000 Ada. The horizontal line denotes equal runtime. Full measurements are in Tables E.5 and E.3.

At the model level, Table 4 measures a full forward pass of the Cats and Dogs ViT on a batch of 64 images, including the surrounding Transformer blocks and classifier. Two-sided ASAP with 64 slices reduces latency from 69.12 to 45.09 ms, a $1.53\times$ speedup, with an accuracy gap of 0.65 percentage points. The one-sided configuration uses 32 slices and final row normalization; it takes 28.10 ms with a 0.94-point accuracy gap. The once-per-model LS calibration takes approximately 24–25 seconds (Appendix F.2).

Table 4: Cats and Dogs accuracy and runtime over three teachers. Forward timing uses GPU-resident batches of 64 images at 224×224; calibration is performed once. Speedup is relative to the teacher. ASAP-LS-0 uses final row normalization.

Model	Accuracy (%)	Calibration (s)	Forward (ms)	Speedup
Sinkhorn teacher ($S = 10$)	81.09 ± 0.61	–	69.12 ± 0.16	$1.00\times$
ASAP-LS ($L = 64$)	80.44 ± 0.39	25.02 ± 0.17	45.09 ± 0.05	$1.53\times$
ASAP-LS-0 ($L = 32$)	80.15 ± 0.69	24.38 ± 0.16	28.10 ± 0.05	$2.46\times$

5.4 Ablation Studies

The construction relies on sliced potentials to describe the query–key geometry and on the analytic transform to recover attention from a single predicted potential. Table 5 examines both choices on IMDb while keeping the teacher fixed. Raw projections and rank displacement produce roughly twice the output error of sliced potentials under the same two-sided reconstruction. A predictor shared across layers remains close to the separate layer-wise fits, whereas predictions of both potentials without the analytic transform have much larger errors and variability. These comparisons favor a compact predictor of one potential combined with the transport normalization.

Table 5: IMDb ablations on 144 held-out batch–layer cases from three $S = 20$ teachers, with 4096 calibration inputs. Shared coefficients use one predictor across layers. Full results are in Table H.3.

Features	Coefficients	Reconstruction	Output RMSE	Attention rel. ℓ_2
Sliced potentials	Per layer	**ASAP**	0.003 ± 0.004	0.035 ± 0.070
Sliced potentials	Shared	ASAP	0.004 ± 0.006	0.051 ± 0.094
Raw projections	Per layer	ASAP	0.006 ± 0.011	0.077 ± 0.167
Rank displacement	Per layer	ASAP	0.006 ± 0.012	0.084 ± 0.177
Predict both potentials without a c-transform			0.297 ± 1.191	2.767 ± 7.326

We use the 32-feature predictor by default to keep prediction inexpensive; richer predictors can reduce the remaining approximation error. On ImageNet, two additional features reduce relative output RMSE from 0.067 to 0.037 under the two-sided transform; Appendix I.5 compares this extension with larger sliced dictionaries, RBF predictors, and an MLP. The fit is less sensitive to calibration size over the ranges tested: on IMDb, errors are unchanged at the reported precision from 512 to 4096 calibration sequences, each supplying many token-level examples. With 512 calibration images, the ImageNet teacher's relative output error increases by 2.8% when we fit on COCO images instead of ImageNet images. Appendix I.4 gives both transfer directions, and Appendix H reports the calibration-size and slice-count studies.

6 Limitations

ASAP retains quadratic query–key computation even with streamed kernels, and its calibration cost is recovered only through repeated inference. Our classification experiments cover encoder models, with the resolution study reaching 785 tokens; the longer kernel benchmarks do not establish task accuracy at those lengths. The CIFAR-100 gap and the gains from richer predictors show that the sliced linear map can leave meaningful approximation error. Changes in the input distribution may also require refitting. The theory describes unmasked transport plans, while the experiments additionally use key-padding masks and row-normalized value mixing where stated. A lower-triangular causal mask permits only the identity among square doubly-stochastic matrices, so autoregressive attention would require a different choice of marginal constraints.

7 Conclusion

ASAP separates the training of Sinkhorn attention from its inference computation. We learn a small map from sliced transport potentials to the trained layer's query potential, then reconstruct attention with a fixed two-sided entropic c-transform. The resulting operator preserves much of the teacher's performance across language and vision tasks without retraining the backbone. Our experiments connect this approximation to lower layer and model latency, while the streamed implementation combines the same learned predictor with reduced intermediate storage. Sliced potentials thus provide a practical way to reuse a trained transport-based attention layer with less online normalization.

AI Use Statement

Generative AI tools were used to assist with writing and editing the manuscript, developing code, preparing experiments, and creating or editing figures. They also assisted with LaTeX formatting. The authors take responsibility for the final manuscript, code, experimental results, and figures.

Reproducibility Statement

The appendix provides proofs, implementation details, experiment settings, and the timing and evaluation protocol. These describe the teacher normalization convention, offline calibration, and online reconstruction used in the reported experiments.

REFERENCES

Brandon Amos, Giulia Luise, Samuel Cohen, and Ievgen Redko. Meta optimal transport. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 791–813, 2023. URL https://proceedings.mlr.press/v202/amos23a.html.

Clément Bonet, Paul Berg, Nicolas Courty, François Septier, Lucas Drumetz, and Minh-Tan Pham. Spherical sliced-wasserstein. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=jXQ0ipgMdU.

Clément Bonet, Lucas Drumetz, and Nicolas Courty. Sliced-wasserstein distances and flows on cartan-hadamard manifolds. *Journal of Machine Learning Research*, 26(32):1–76, 2025a. URL https://jmlr.org/papers/v26/24-0359.html.

Clément Bonet, Kimia Nadjahi, Thibault Séjourné, Kilian Fatras, and Nicolas Courty. Slicing unbalanced optimal transport. *Transactions on Machine Learning Research*, 2025b. URL https://openreview.net/forum?id=AjJTg5M0r8.

Jannis Born, Filip Skogh, Kahn Rhrissorrakrai, Filippo Utro, Nico Wagner, and Aleksandros Sobczyk. Quantum doubly stochastic transformers. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=0EILv1HcmG.

Laetitia Chapel, Romain Tavenard, and Samuel Vaiter. Differentiable generalized sliced wasserstein plans. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2505.22049.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Łukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, 2013.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html.

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. *arXiv preprint arXiv:2103.03404*, 2021. URL https://arxiv.org/abs/2103.03404.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.11929.

Doron Haviv, Russell Zhang Kunes, Thomas Dougherty, Cassandra Burdziak, Tal Nawy, Anna Gilbert, and Dana Pe'Er. Wasserstein wormhole: Scalable optimal transport distance with transformer. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17697–17718, 2024. URL https://proceedings.mlr.press/v235/haviv24a.html.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=8CR8OFUUE6sl.

Michela Lapenna, Rita Fioresi, and Bahman Gharesifard. Sinkhorn doubly stochastic attention rank decay analysis. *arXiv preprint arXiv:2604.07925*, 2026. URL https://arxiv.org/abs/2604.07925.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common objects in context. *arXiv preprint arXiv:1405.0312*, 2014. URL https://arxiv.org/abs/1405.0312.

Xinran Liu, Elaheh Akbari, Rocio Diaz Martin, Navid NaderiAlizadeh, and Soheil Kolouri. Efficient transferable optimal transport via min-sliced transport plans. *arXiv preprint arXiv:2511.19741*, 2025a. URL https://arxiv.org/abs/2511.19741.

Xinran Liu, Rocio Diaz Martin, Yikun Bai, Ashkan Shahbazi, Matthew Thorpe, Akram Aldroubi, and Soheil Kolouri. Expected sliced transport plans. In *International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=P7O1Vt1BdU.

Guillaume Mahey, Laetitia Chapel, Gilles Gasso, Clément Bonet, and Nicolas Courty. Fast optimal transport through sliced generalized wasserstein geodesics. In *Advances in Neural Information Processing Systems*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6f1346bac8b02f76a631400e2799b24b-Abstract-Conference.html.

Khai Nguyen. An introduction to sliced optimal transport. *arXiv preprint arXiv:2508.12519*, 2025. URL https://arxiv.org/abs/2508.12519.

Khai Nguyen. Sliced-regularized optimal transport. *arXiv preprint arXiv:2604.23944*, 2026. URL https://arxiv.org/abs/2604.23944.

Khai Nguyen and Nhat Ho. Revisiting sliced wasserstein on images: From vectorization to convolution. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/712a2cefb5c84274061e2dd4f6464f31-Abstract-Conference.html.

Khai Nguyen and Nhat Ho. Energy-based sliced wasserstein distance. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/3a23caeb904c822575fa56fb114ca499-Abstract-Conference.html.

Khai Nguyen, Nicola Bariletto, and Nhat Ho. Quasi-monte carlo for 3d sliced wasserstein. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Wd47f7HEXg.

Khai Nguyen, Hai Nguyen, and Nhat Ho. Fast estimation of wasserstein distances via regression on sliced wasserstein distances. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=oa7L4vcJ77.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015. doi: 10.1007/s11263-015-0816-y. URL https://arxiv.org/abs/1409.0575.

Michael E. Sander, Pierre Ablin, Mathieu Blondel, and Gabriel Peyré. Sinkformers: Transformers with doubly stochastic attention. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 3515–3530, 2022. URL https://proceedings.mlr.press/v151/sander22a.html.

Ashkan Shahbazi, Elaheh Akbari, Darian Salehi, Xinran Liu, Navid NaderiAlizadeh, and Soheil Kolouri. Espformer: Doubly-stochastic attention with expected sliced transport plans. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 54186–54202, 2025a. URL https://proceedings.mlr.press/v267/shahbazi25a.html.

Ashkan Shahbazi, Chayne Thrash, Yikun Bai, Keaton Hamm, Navid NaderiAlizadeh, and Soheil Kolouri. Lotformer: Doubly-stochastic linear attention via low-rank optimal transport. *arXiv preprint arXiv:2509.23436*, 2025b. URL https://arxiv.org/abs/2509.23436.

Eloi Tanguy, Laetitia Chapel, and Julie Delon. Sliced optimal transport plans. *arXiv preprint arXiv:2508.01243*, 2025. URL https://arxiv.org/abs/2508.01243.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9438–9447, 2020. URL https://proceedings.mlr.press/v119/tay20a.html.

James Thornton and Marco Cuturi. Rethinking initialization of the sinkhorn algorithm. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 8682–8698, 2023. URL https://proceedings.mlr.press/v206/thornton23a.html.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10347–10357, 2021. URL https://proceedings.mlr.press/v139/touvron21a.html.

Huy Tran, Yikun Bai, Abihith Kothapalli, Ashkan Shahbazi, Xinran Liu, Rocio P. Diaz Martin, and Soheil Kolouri. Stereographic spherical sliced Wasserstein distances. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 48494–48564, 2024. URL https://proceedings.mlr.press/v235/tran24a.html.

Minh-Phuc Truong and Khai Nguyen. Amortized optimal transport from sliced potentials. *arXiv preprint arXiv:2604.15114*, 2026. URL https://arxiv.org/abs/2604.15114.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1706.03762.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. URL https://arxiv.org/abs/2006.04768.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*, volume 35, pp. 14138–14148, 2021. URL https://aaai.org/papers/14138-nystromformer-a-nystrom-based-algorithm-for-approximating-self-attention/.

Felix X.-F. Ye, Xingjie Li, An Yu, Ming-Ching Chang, Linsong Chu, and Davis Wertheimer. Flashsinkhorn: Io-aware entropic optimal transport. *arXiv preprint arXiv:2602.03067*, 2026. URL https://arxiv.org/abs/2602.03067.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential transformer. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OvoCm1gGhN.

A Notation and Conventions

Symbol	Convention
Basic objects.	
$\mathbf{1}_m$	all-ones vector in $\mathbb{R}^m$
$\mathrm{Diag}(a)$	diagonal matrix with diagonal entries a_i
Δ_m	$\{p \in \mathbb{R}^m_+ : \mathbf{1}_m^\top p = 1\}$
Δ_m^+	$\{p \in \Delta_m : p_i > 0 \; \forall i\}$

continued on next page

Symbol	Convention
$\langle C, P\rangle$	$\sum_{ij} C_{ij}P_{ij}$ for matrices of the same size
$\Pi_0(z)$	$z - \frac{1}{N}(\mathbf{1}_N^\top z)\mathbf{1}_N$, the zero-mean projection
Tokens and attention.	
N	sequence length after padding or patch/token construction
d_h, d_v	head dimension and value dimension
Q, K, V	matrices in $\mathbb{R}^{N\times d_h}$, $\mathbb{R}^{N\times d_h}$, and $\mathbb{R}^{N\times d_v}$
q_i, k_j, v_j	rows of Q, K, V
S_{ij}	scaled dot-product score $\langle q_i, k_j\rangle/\sqrt{d_h}$
C_{ij}	quadratic cost $\|q_i - k_j\|_2^2/(2\sqrt{d_h})$ used in the main text
$\rho_i(Q), \kappa_j(K)$	score-to-cost shift terms $\|q_i\|_2^2/(2\sqrt{d_h})$ and $\|k_j\|_2^2/(2\sqrt{d_h})$
Transport and Sinkhorn quantities.	
μ, ν	uniform marginals $\frac{1}{N}\mathbf{1}_N$ in the unmasked square setting
$\mathcal{U}(\mu,\nu)$	$\{P \in \mathbb{R}_+^{N\times N} : P\mathbf{1}_N = \mu,\ P^\top\mathbf{1}_N = \nu\}$
P	coupling in transport units
A	attention-scale matrix $A = NP$ used for value mixing
f, g	source and key Sinkhorn duals, defined up to additive shifts $(f,g) \mapsto (f + c\mathbf{1}_N, g - c\mathbf{1}_N)$
ε	entropic regularization parameter or its equivalent score-kernel scale
$T_K^\nu(f)$	key-side entropic c-transform applied to a source dual f
$T_Q^\mu(g)$	source-side entropic c-transform applied to a key dual g
T	finite Sinkhorn budget used by the teacher
S	finite Sinkhorn budget used by an iterative comparison baseline
ASAP fitting quantities.	
$\Theta = \{\theta_\ell\}_{\ell=1}^L$	fixed slice directions used by an ASAP layer
$a_i^{(\ell)}, b_j^{(\ell)}$	one-dimensional projections of scaled queries and keys along θ_ℓ
$\tilde{f}^{(\ell)}$	centered one-dimensional source potential from slice ℓ
$X_\Theta(Q,K)$	matrix in $\mathbb{R}^{N\times L}$ whose columns are $\tilde{f}^{(\ell)}(Q,K)$
M	number of unlabeled fit examples used by an offline projection
λ	regularization parameter for an offline projection
ω	ASAP coefficient vector in $\mathbb{R}^L$
$\hat{f}_\omega$	centered predicted source dual $\Pi_0(X_\Theta(Q,K)\omega)$
ASAP-KL	ASAP with the convex KL fit through the key-side c-transform
ASAP-LS	ASAP with the closed-form least-squares dual fit
ASAP-0	one-sided ablation using $\hat{g}_\omega^0 = T_K^\nu(\hat{f}_\omega)$
ASAP	default compiled layer using the two-sided transform in Equation 9
Masks, errors, and timing.	
$J(x)$	active key positions for a padded text input x
ν^J	masked key target with $\nu_j^J = 1/N$ for $j \in J(x)$ and $\nu_j^J = 0$ otherwise
$e_{\mathrm{row}}(A)$	$N^{-1}\sum_i \lvert\sum_j A_{ij} - 1\rvert$ in attention units
$e_{\mathrm{col}}(A; J)$	$\lvert J\rvert^{-1}\sum_{j\in J}\lvert\sum_i A_{ij} - 1\rvert$ for text batches, or $N^{-1}\sum_j \lvert\sum_i A_{ij} - 1\rvert$ without padding
layer-batch case	one held-out minibatch evaluated at one frozen attention layer, with heads included in the average unless stated otherwise
fit time	teacher extraction and offline fitting; reported outside online forward latency

B PROOFS

Exact row/column-shift invariance.

Proposition B.1. *Fix marginals μ, ν and $\varepsilon > 0$. If $\widetilde{C}_{ij} = C_{ij} + a_i + b_j$ for some $a, b \in \mathbb{R}^N$, then C and $\widetilde{C}$ induce the same entropic optimal transport plan on $\mathcal{U}(\mu,\nu)$.*

Proof. Let $J_C(P) = \langle C, P\rangle + \varepsilon \sum_{i,j} P_{ij}(\log P_{ij} - 1)$ denote the entropic OT objective over $P \in \mathcal{U}(\mu, \nu)$. Suppose P is feasible. Fixed marginals make row-only and column-only additions constant across the feasible set. Indeed, if $\widetilde{C}_{ij} = C_{ij} + a_i + b_j$, we have

$$J_{\widetilde{C}}(P) = J_C(P) + \sum_{i,j}(a_i + b_j)P_{ij} = J_C(P) + a^\top \mu + b^\top \nu. \tag{B.1}$$

The added term $a^\top \mu + b^\top \nu$ does not depend on P. The two objectives are therefore the same up to a constant on the transport polytope, so they choose the same minimizers. For the attention kernel, we can take

$$\widetilde{C}_{ij} = -S_{ij}, \qquad C_{ij} = \frac{1}{2\sqrt{d_h}}\|q_i - k_j\|_2^2, \tag{B.2}$$

so that

$$-C_{ij} = S_{ij} - \frac{1}{2\sqrt{d_h}}\|q_i\|_2^2 - \frac{1}{2\sqrt{d_h}}\|k_j\|_2^2. \tag{B.3}$$

So the quadratic-cost and scaled-dot-product formulations are the same up to row and column terms. For the exact two-marginal entropic problem, those terms are invisible to the optimizer, and the induced plan is the same. □

For exact entropic OT, the minimizer is unchanged by row and column shifts of the cost. With finitely many Sinkhorn updates, the output is also tied to the chosen kernel, initialization, and update order. ASAP therefore fits and reconstructs under the same finite-budget convention.

Validity of the one-dimensional sliced potential.

Proposition B.2. *Let $a_{(1)} \le \cdots \le a_{(N)}$ and $b_{(1)} \le \cdots \le b_{(N)}$ be sorted one-dimensional supports with uniform masses. Define*

$$\phi_1 = 0, \qquad \phi_r = \sum_{t=1}^{r-1} b_{(t)}(a_{(t+1)} - a_{(t)}), \qquad r \ge 2, \tag{B.4}$$

and $f_{(r)} = \frac{1}{2}a_{(r)}^2 - \phi_r$. The vector f is a valid source Kantorovich potential for the monotone optimal matching under the cost $c(a, b) = \frac{1}{2}(a - b)^2$.

Proof. Assume first that the source locations are all distinct. We want a dual pair that is tight exactly on the sorted matching. Extend ϕ to a piecewise-linear function with $\phi(a_{(r)}) = \phi_r$ and slope $b_{(r)}$ on each interval $[a_{(r)}, a_{(r+1)}]$, using outer rays with slopes $b_{(1)}$ and $b_{(N)}$. The sorted target values are nondecreasing, so these slopes are nondecreasing and ϕ is convex. We write $\phi^\star$ for its convex conjugate and define

$$h_{(s)} = \frac{1}{2}b_{(s)}^2 - \phi^\star(b_{(s)}). \tag{B.5}$$

By Fenchel's inequality, $\phi(a) + \phi^\star(b) \ge ab$ for all a, b. Therefore, for every source index r and target index s, we have

$$f_{(r)} + h_{(s)} \le \frac{1}{2}a_{(r)}^2 + \frac{1}{2}b_{(s)}^2 - a_{(r)}b_{(s)} = \frac{1}{2}(a_{(r)} - b_{(s)})^2. \tag{B.6}$$

This is the dual-feasibility inequality for the quadratic-cost transport problem. On a matched pair $(a_{(r)}, b_{(r)})$, our slope convention places $b_{(r)}$ in $\partial\phi(a_{(r)})$, so Fenchel equality holds and the dual bound is tight. The sorted matching is primal feasible. After we average these equalities over the uniform atoms, the primal cost and dual value agree, so the sorted matching is optimal and f is a valid source potential.

If several source locations are tied, the zero gaps leave ϕ_r unchanged across the tied block. The sorted target values assigned to that block lie in the corresponding subgradient interval, so the same equality argument applies after fixing any deterministic monotone tie rule. □

One-sided entropic c-transform.

Proposition B.3. *Fix a positive key marginal ν and $f \in \mathbb{R}^N$, and define*

$$g = T_K^\nu(f), \qquad P_{fij} = \exp\left(\frac{-C_{ij} + f_i + g_j}{\varepsilon}\right). \tag{B.7}$$

The plan then satisfies $P_f^\top \mathbf{1}_N = \nu$. Moreover, for any constant $c \in \mathbb{R}$,

$$T_K^\nu(f + c\mathbf{1}_N) = T_K^\nu(f) - c\mathbf{1}_N, \qquad P_{f+c\mathbf{1}_N} = P_f. \tag{B.8}$$

Proof. We exponentiate the definition of the key-side c-transform to get

$$\exp\left(\frac{g_j}{\varepsilon}\right) = \frac{\nu_j}{\sum_{i=1}^N \exp\left(\frac{-C_{ij}+f_i}{\varepsilon}\right)}. \tag{B.9}$$

The c-transform was chosen so that this denominator cancels inside each key column. For key j, we have

$$\sum_{i=1}^N P_{fij} = \exp\left(\frac{g_j}{\varepsilon}\right) \sum_{i=1}^N \exp\left(\frac{-C_{ij} + f_i}{\varepsilon}\right) = \nu_j, \tag{B.10}$$

which proves $P_f^\top \mathbf{1}_N = \nu$.

Now let $c \in \mathbb{R}$. The constant shift in the source dual appears as a common factor inside the logarithm in $T_K^\nu(f + c\mathbf{1}_N)_j$, so

$$T_K^\nu(f + c\mathbf{1}_N)_j = \varepsilon \log \nu_j - \varepsilon \log\left(\exp(c/\varepsilon) \sum_{i=1}^N \exp\left(\frac{-C_{ij} + f_i}{\varepsilon}\right)\right) = T_K^\nu(f)_j - c. \tag{B.11}$$

When we place this shifted key dual back into the plan, the two constants cancel:

$$P_{f+c\mathbf{1}_N ij} = \exp\left(\frac{-C_{ij} + (f_i + c) + (g_j - c)}{\varepsilon}\right) = P_{fij}, \tag{B.12}$$

which proves additive-shift invariance. □

Proposition B.4. *Fix C, $\varepsilon > 0$, and a positive key marginal ν. For source duals $f, \widehat{f} \in \mathbb{R}^N$, let P_f and $P_{\widehat{f}}$ be the key-normalized plans obtained from $T_K^\nu(f)$ and $T_K^\nu(\widehat{f})$. Under this notation, we have*

$$P_{\widehat{f}} = P_f \quad \Longleftrightarrow \quad \widehat{f} - f \in \operatorname{span}\{\mathbf{1}_N\}. \tag{B.13}$$

In particular, if f and $\widehat{f}$ are centered, then $P_{\widehat{f}} = P_f$ if and only if $\widehat{f} = f$.

Proof. One direction is exactly the additive-shift invariance in Proposition B.3. We prove the converse. Suppose $P_{\widehat{f}} = P_f$. Once the key marginal is fixed, each key column is a normalized distribution over queries. For each key j, the key-side c-transform lets us write

$$P_{fij} = \nu_j \frac{\exp((-C_{ij} + f_i)/\varepsilon)}{\sum_r \exp((-C_{rj} + f_r)/\varepsilon)} \tag{B.14}$$

and the same formula holds with f replaced by $\widehat{f}$. Since $\nu_j > 0$ and the two plans are equal, the normalized distributions within each key column are equal. Therefore, for every i and j,

$$\frac{\exp((-C_{ij} + \widehat{f}_i)/\varepsilon)}{\exp((-C_{ij} + f_i)/\varepsilon)} \tag{B.15}$$

is independent of i within that column. The cost terms cancel, so $\exp((\widehat{f}_i - f_i)/\varepsilon)$ is independent of i. Hence $\widehat{f} - f = c\mathbf{1}_N$ for some constant c. If both vectors are centered, the constant has mean zero and $c = 0$. □

Proposition B.5. *Fix C, $\varepsilon > 0$, a source dual $f \in \mathbb{R}^N$, and a positive key marginal ν. Define*

$$R^f_{ij} = \exp\left(\frac{-C_{ij} + f_i}{\varepsilon}\right) \tag{B.16}$$

and

$$\mathcal{K}_\nu = \{P \in \mathbb{R}^{N\times N}_+ : P^\top \mathbf{1}_N = \nu\}. \tag{B.17}$$

We use the generalized KL divergence below because R^f is unnormalized. For nonnegative P and positive R, use the generalized KL divergence

$$D^{\mathrm{gen}}_{\mathrm{KL}}(P\|R) = \sum_{i,j}\left[P_{ij}\log\frac{P_{ij}}{R_{ij}} - P_{ij} + R_{ij}\right]. \tag{B.18}$$

The key-normalized Gibbs plan $P_f = \mathcal{C}_\nu(f; C)$ is the unique minimizer of

$$\min_{P\in\mathcal{K}_\nu} D^{\mathrm{gen}}_{\mathrm{KL}}(P\|R^f). \tag{B.19}$$

Proof. The key constraint is columnwise, so the projection can be solved one key at a time. For a fixed key j, we minimize

$$\sum_i\left[p_i\log\frac{p_i}{R^f_{ij}} - p_i + R^f_{ij}\right] \tag{B.20}$$

subject to $p_i \geq 0$ and $\sum_i p_i = \nu_j$. The objective is strictly convex on the positive orthant. With a Lagrange multiplier λ_j for the column-mass constraint, stationarity requires

$$\log\frac{p_i}{R^f_{ij}} + \lambda_j = 0. \tag{B.21}$$

Since $R^f_{ij} > 0$ and $\nu_j > 0$, the minimizer lies in the positive orthant, so this stationarity condition characterizes it. We have $p_i = R^f_{ij}\exp(-\lambda_j)$, and the value of λ_j is determined by the column sum:

$$p_i = \nu_j\frac{R^f_{ij}}{\sum_r R^f_{rj}} = \exp\left(\frac{-C_{ij} + f_i + T^\nu_K(f)_j}{\varepsilon}\right). \tag{B.22}$$

We have recovered the key-side c-transform. Strict convexity on each positive column ensures uniqueness, and the columnwise minimizers assemble into the unique key-normalized plan. □

We write $\mathcal{C}_\nu(f; C)$ for this key-normalized Gibbs plan. This is the plan used by ASAP-0 and by the convex KL calibration objective below.

Column-ending two-sided entropic c-transform.

Proposition B.6. *Fix positive marginals $\mu, \nu \in \mathbb{R}^N_+$ with $\mathbf{1}_N^\top\mu = \mathbf{1}_N^\top\nu = 1$. For a source dual $f \in \mathbb{R}^N$, define*

$$\begin{aligned} g^0 &= T^\nu_K(f), \\ f^+ &= T^\mu_Q(g^0), \\ g^+ &= T^\nu_K(f^+), \end{aligned} \tag{B.23}$$

where

$$T^\mu_Q(g)_i = \varepsilon\log\mu_i - \varepsilon\log\sum_{j=1}^N\exp\left(\frac{-C_{ij} + g_j}{\varepsilon}\right). \tag{B.24}$$

Define the intermediate and final plans

$$R^f_{ij} = \exp\left(\frac{-C_{ij} + f^+_i + g^0_j}{\varepsilon}\right), \qquad P^{\leftrightarrow}_{f,ij} = \exp\left(\frac{-C_{ij} + f^+_i + g^+_j}{\varepsilon}\right). \tag{B.25}$$

Then

$$R^f\mathbf{1}_N = \mu, \qquad \left(P^{\leftrightarrow}_f\right)^\top\mathbf{1}_N = \nu. \tag{B.26}$$

Moreover, for any constant $c \in \mathbb{R}$,

$$P^{\leftrightarrow}_{f+c\mathbf{1}_N} = P^{\leftrightarrow}_f. \tag{B.27}$$

Proof. We choose the row-side transform so that each query row of the intermediate plan has the prescribed mass. By definition,

$$\exp\left(\frac{f_i^+}{\varepsilon}\right) = \frac{\mu_i}{\sum_j \exp\left(\frac{-C_{ij}+g_j^0}{\varepsilon}\right)}. \tag{B.28}$$

We substitute this expression into the row sum of R^f and get

$$\sum_j R_{ij}^f = \exp\left(\frac{f_i^+}{\varepsilon}\right) \sum_j \exp\left(\frac{-C_{ij}+g_j^0}{\varepsilon}\right) = \mu_i. \tag{B.29}$$

We then apply the same column argument as Proposition B.3, now with the updated source dual f^+, and obtain

$$\sum_i P_{f,ij}^{\leftrightarrow} = \nu_j. \tag{B.30}$$

For shift invariance, let $f' = f + c\mathbf{1}_N$. Proposition B.3 yields $g^{0\prime} = g^0 - c\mathbf{1}_N$. The source-side transform moves by the opposite amount:

$$T_Q^\mu(g^0 - c\mathbf{1}_N) = T_Q^\mu(g^0) + c\mathbf{1}_N, \tag{B.31}$$

so $f^{+\prime} = f^+ + c\mathbf{1}_N$. We apply Proposition B.3 one more time and get $g^{+\prime} = g^+ - c\mathbf{1}_N$. The constants cancel inside the exponential expression,

$$\exp\left(\frac{-C_{ij} + f_i^{+\prime} + g_j^{+\prime}}{\varepsilon}\right) = \exp\left(\frac{-C_{ij} + f_i^+ + g_j^+}{\varepsilon}\right), \tag{B.32}$$

which proves $P_{f+c\mathbf{1}_N}^{\leftrightarrow} = P_f^{\leftrightarrow}$. □

The proposition states a row-marginal property for the intermediate plan R^f and a column-marginal property for the returned plan $P_f^{\leftrightarrow}$. It does not assert that $P_f^{\leftrightarrow}$ satisfies both marginals exactly. The row-ending transform used for teachers whose last Sinkhorn step normalizes rows returns R^f instead.

Proposition B.7. *Under the notation of Proposition B.6, set*

$$\delta_f = \|g^+ - g^0\|_\infty. \tag{B.33}$$

Then the row error of the final column-marginal plan obeys

$$\left\|P_f^{\leftrightarrow}\mathbf{1}_N - \mu\right\|_1 \le \left\|P_f^{\leftrightarrow} - R^f\right\|_1 \le \exp(\delta_f/\varepsilon) - 1. \tag{B.34}$$

The matrix norm in the middle term is entrywise ℓ_1.

Proof. By Proposition B.6, the intermediate plan satisfies $R^f\mathbf{1}_N = \mu$. Row summation cannot increase entrywise ℓ_1 distance, so

$$\left\|P_f^{\leftrightarrow}\mathbf{1}_N - \mu\right\|_1 = \left\|\left(P_f^{\leftrightarrow} - R^f\right)\mathbf{1}_N\right\|_1 \le \left\|P_f^{\leftrightarrow} - R^f\right\|_1. \tag{B.35}$$

Next compare the final plan with the intermediate plan. They are related by a columnwise factor:

$$P_{f,ij}^{\leftrightarrow} = R_{ij}^f \exp\left(\frac{g_j^+ - g_j^0}{\varepsilon}\right). \tag{B.36}$$

Since $|g_j^+ - g_j^0| \le \delta_f$, we have

$$\left|\exp\left(\frac{g_j^+ - g_j^0}{\varepsilon}\right) - 1\right| \le \exp(\delta_f/\varepsilon) - 1. \tag{B.37}$$

We multiply this uniform bound by $\sum_{i,j} R_{ij}^f = 1$, which proves the claim. □

Dual-error stability.

Proposition B.8. *Fix C, $\varepsilon > 0$, and a positive key marginal ν. For $f, \widehat{f} \in \mathbb{R}^N$, define*

$$g = T_K^\nu(f), \qquad \widehat{g} = T_K^\nu(\widehat{f}), \tag{B.38}$$

and let P_f and $P_{\widehat{f}}$ be the corresponding reconstructed plans. If

$$\|\widehat{f} - f\|_\infty \le \alpha, \tag{B.39}$$

then

$$\|\widehat{g} - g\|_\infty \le \alpha \tag{B.40}$$

and, for every i, j,

$$e^{-2\alpha/\varepsilon} P_{f,ij} \le P_{\widehat{f},ij} \le e^{2\alpha/\varepsilon} P_{f,ij}. \tag{B.41}$$

Proof. Let $\Delta f = \widehat{f} - f$. Fix a key j. The two key-side c-transforms change only through the vector inside one log-sum-exp. Log-sum-exp is 1-Lipschitz in ℓ_∞, and the two input vectors are at most $\|\Delta f\|_\infty/\varepsilon$ apart. Therefore

$$|\widehat{g}_j - g_j| \le \varepsilon \cdot \frac{\|\Delta f\|_\infty}{\varepsilon} = \|\Delta f\|_\infty \le \alpha. \tag{B.42}$$

This holds for every key, so the whole key dual moves by at most α, and $\|\widehat{g} - g\|_\infty \le \alpha$.

Now compare the two plans entry by entry. Their ratio contains only the source-dual error and the induced key-dual error:

$$\frac{P_{\widehat{f},ij}}{P_{f,ij}} = \exp\left(\frac{(\widehat{f}_i - f_i) + (\widehat{g}_j - g_j)}{\varepsilon}\right). \tag{B.43}$$

Because $|\widehat{f}_i - f_i| \le \alpha$ and $|\widehat{g}_j - g_j| \le \alpha$, the exponent lies in the interval $[-2\alpha/\varepsilon,\, 2\alpha/\varepsilon]$. Hence

$$e^{-2\alpha/\varepsilon} \le \frac{P_{\widehat{f},ij}}{P_{f,ij}} \le e^{2\alpha/\varepsilon}, \tag{B.44}$$

which is equivalent to

$$e^{-2\alpha/\varepsilon} P_{f,ij} \le P_{\widehat{f},ij} \le e^{2\alpha/\varepsilon} P_{f,ij}. \tag{B.45}$$

□

Because the reconstruction is invariant to additive shifts, the plan-ratio bound also holds with α replaced by $\inf_{c\in\mathbb{R}} \|\widehat{f} - f - c\mathbf{1}_N\|_\infty$. The key-dual bound itself is a statement about a fixed representative of the source dual. In the main construction, both teacher and predicted source duals are centered before fitting.

Proposition B.9. *Let $P_f^{\leftrightarrow}$ and $P_{\widehat{f}}^{\leftrightarrow}$ be the two-sided plans from Proposition B.6. If*

$$\|\widehat{f} - f\|_\infty \le \alpha, \tag{B.46}$$

then every dual produced by the two-sided transform moves by at most α:

$$\|\widehat{g}^0 - g^0\|_\infty \le \alpha, \qquad \|\widehat{f}^+ - f^+\|_\infty \le \alpha, \qquad \|\widehat{g}^+ - g^+\|_\infty \le \alpha. \tag{B.47}$$

Consequently, for every i, j,

$$e^{-2\alpha/\varepsilon} P^{\leftrightarrow}_{f,ij} \le P^{\leftrightarrow}_{\widehat{f},ij} \le e^{2\alpha/\varepsilon} P^{\leftrightarrow}_{f,ij}. \tag{B.48}$$

Proof. Proposition B.8 yields the key-side bound $\|\widehat{g}^0 - g^0\|_\infty \le \alpha$. The same log-sum-exp Lipschitz argument applies to the source-side transform T_Q^μ, so $\|\widehat{f}^+ - f^+\|_\infty \le \|\widehat{g}^0 - g^0\|_\infty \le \alpha$. We apply Proposition B.8 again to f^+ and $\widehat{f}^+$ and get $\|\widehat{g}^+ - g^+\|_\infty \le \alpha$.

We now compare the two final plans:

$$\frac{P^{\leftrightarrow}_{\widehat{f},ij}}{P^{\leftrightarrow}_{f,ij}} = \exp\left(\frac{(\widehat{f}_i^+ - f_i^+) + (\widehat{g}_j^+ - g_j^+)}{\varepsilon}\right). \tag{B.49}$$

Both terms in the numerator are bounded in absolute value by α, which proves the displayed multiplicative bound. □

The proposition compares outputs of the same two-sided map. When the finite teacher plan is defined by the one-sided final closure, default ASAP has an additional finite-correction bias term such as $\|P^{\leftrightarrow}_{f^{(T)}} - P^{(T)}\|$. ASAP-0 has zero such term under exact prediction in the column-ending final-closure case.

Proposition B.10. *Fix C, $\varepsilon > 0$, and a positive key marginal ν with $\sum_j \nu_j = 1$. Let $P = P_f$ and $\widehat{P} = P_{\widehat{f}}$ be the key-normalized Gibbs plans induced by source duals f and $\widehat{f}$ under the same cost and key marginal. For each key j, set $\pi_j(i) = P_{ij}/\nu_j$, $\widehat{\pi}_j(i) = \widehat{P}_{ij}/\nu_j$, and $\Delta_i = \widehat{f}_i - f_i$. For $t \in [0,1]$, define the tilted column distribution*

$$\pi_{j,t}(i) = \frac{\pi_j(i)\exp(t\Delta_i/\varepsilon)}{\sum_r \pi_j(r)\exp(t\Delta_r/\varepsilon)}. \tag{B.50}$$

The reverse KL identity is

$$D_{\mathrm{KL}}(\widehat{P}\|P) = \sum_j \nu_j \int_0^1 t\ \mathrm{Var}_{\pi_{j,t}}(\Delta/\varepsilon)\, dt. \tag{B.51}$$

The forward direction satisfies the companion identity

$$D_{\mathrm{KL}}(P\|\widehat{P}) = \sum_j \nu_j \int_0^1 (1-t)\ \mathrm{Var}_{\pi_{j,t}}(\Delta/\varepsilon)\, dt. \tag{B.52}$$

The two identities use two weights on the same interpolation; the common quadratic bound below comes from a uniform ceiling on the variance over t. Consequently, if $\mathrm{Var}_{\pi_{j,t}}(\Delta) \le \sigma_j^2$ for all $t \in [0,1]$, then

$$D_{\mathrm{KL}}(\widehat{P}\|P),\ D_{\mathrm{KL}}(P\|\widehat{P}) \le \frac{1}{2\varepsilon^2}\sum_j \nu_j \sigma_j^2. \tag{B.53}$$

If, additionally, $\pi_{j,t}(i) \le \beta/N$ for all i, j, t, then

$$D_{\mathrm{KL}}(\widehat{P}\|P),\ D_{\mathrm{KL}}(P\|\widehat{P}) \le \frac{\beta}{2\varepsilon^2 N}\|\widehat{f} - f\|_2^2. \tag{B.54}$$

The constant β measures column sharpness: $\beta = 1$ corresponds to uniform column conditionals, while sharper column conditionals give larger β.

Proof. For each key j, the key-side c-transform turns the new column conditional into an exponential tilt of the old one. In particular,

$$\widehat{\pi}_j(i) = \frac{\pi_j(i)\exp(\Delta_i/\varepsilon)}{\sum_r \pi_j(r)\exp(\Delta_r/\varepsilon)}. \tag{B.55}$$

Let

$$A_j(t) = \log \sum_i \pi_j(i)\exp(t\Delta_i/\varepsilon). \tag{B.56}$$

This is the log-partition function along the interpolation. Its first derivative is the tilted mean of Δ/ε, and its second derivative is the tilted variance:

$$A_j'(t) = \sum_i \pi_{j,t}(i)\Delta_i/\varepsilon, \qquad A_j''(t) = \mathrm{Var}_{\pi_{j,t}}(\Delta/\varepsilon). \tag{B.57}$$

At $t = 1$, the interpolation reaches $\widehat{\pi}_j$. We can write the reverse column KL as

$$D_{\mathrm{KL}}(\widehat{\pi}_j\|\pi_j) = \sum_i \pi_{j,1}(i)\,(\Delta_i/\varepsilon - A_j(1)) = A_j'(1) - A_j(1). \tag{B.58}$$

Since $A_j(0) = 0$, this becomes

$$A_j'(1) - A_j(1) = A_j'(1) - \int_0^1 A_j'(s)\, ds = \int_0^1 tA_j''(t)\, dt. \tag{B.59}$$

The forward column KL uses the same interpolation, with the weight reversed:

$$D_{\mathrm{KL}}(\pi_j \| \widehat{\pi}_j) = \sum_i \pi_j(i)\,(A_j(1) - \Delta_i/\varepsilon) = A_j(1) - A_j'(0). \tag{B.60}$$

Since $A_j(1) = \int_0^1 A_j'(s)\,ds$, we obtain

$$A_j(1) - A_j'(0) = \int_0^1 (1-t) A_j''(t)\,dt. \tag{B.61}$$

Returning to plan scale, $P_{ij} = \nu_j \pi_j(i)$ and $\widehat{P}_{ij} = \nu_j \widehat{\pi}_j(i)$. Therefore $D_{\mathrm{KL}}(\widehat{P}\|P) = \sum_j \nu_j D_{\mathrm{KL}}(\widehat{\pi}_j \| \pi_j)$ and $D_{\mathrm{KL}}(P\|\widehat{P}) = \sum_j \nu_j D_{\mathrm{KL}}(\pi_j \| \widehat{\pi}_j)$. Summing over columns proves both identities. The first bounds follow from the variance ceiling and the identities $\int_0^1 t\,dt = \int_0^1 (1-t)\,dt = 1/2$. For the last bounds, the density condition implies

$$\mathrm{Var}_{\pi_{j,t}}(\Delta) \le \sum_i \pi_{j,t}(i)\Delta_i^2 \le \frac{\beta}{N}\|\Delta\|_2^2, \tag{B.62}$$

and $\sum_j \nu_j = 1$. We plug this estimate into the previous bound to finish the proof. □

One-sided KL calibration. The next propositions concern the key-normalized plan $\mathcal{C}_\nu(f; C)$. They justify the convex ASAP-KL calibration surrogate for the one-sided plan; they do not claim convexity of the full two-sided ASAP inference map.

Proposition B.11. *For fit examples $m = 1, \dots, M$, let*

$$F_m(f) = \varepsilon \sum_j \nu_j \log \sum_i \exp\left(\frac{-C_{ij}^{(m)} + f_i}{\varepsilon}\right), \qquad \widehat{P}_\omega^{0,(m)} = \mathcal{C}_\nu(X_m\omega; C^{(m)}), \tag{B.63}$$

and let $r^{(T,m)} = P^{(T,m)}\mathbf{1}_N$. Assume $P^{(T,m)\top}\mathbf{1}_N = \nu$ for each fit example. For $\lambda > 0$, define

$$J_{\mathrm{KL}}(\omega) = \sum_{m=1}^M \left[F_m(X_m\omega) - \langle r^{(T,m)}, X_m\omega\rangle\right] + \frac{\lambda}{2}\|\omega\|_2^2. \tag{B.64}$$

The objective J_{KL} is strongly convex and has a unique minimizer $\omega^\star$. This minimizer satisfies

$$\sum_{m=1}^M X_m^\top \left(\widehat{P}_{\omega^\star}^{0,(m)}\mathbf{1}_N - r^{(T,m)}\right) = -\lambda\omega^\star. \tag{B.65}$$

Moreover, for $f_m = X_m\omega$ and $\pi_j^{(m)}(i) = \mathcal{C}_\nu(f_m; C^{(m)})_{ij}/\nu_j$,

$$\nabla_\omega^2 J_{\mathrm{KL}}(\omega) = \sum_{m=1}^M X_m^\top \left[\frac{1}{\varepsilon}\sum_j \nu_j \left(\mathrm{Diag}(\pi_j^{(m)}) - \pi_j^{(m)}(\pi_j^{(m)})^\top\right)\right] X_m + \lambda I_L \succeq \lambda I_L. \tag{B.66}$$

Proof. The convexity comes from the log-sum-exp terms. Each F_m is composed with the linear map $\omega \mapsto X_m\omega$, so $F_m(X_m\omega)$ is convex in ω. The term $-\langle r^{(T,m)}, X_m\omega\rangle$ is affine, and $\lambda\|\omega\|_2^2/2$ is λ-strongly convex. Therefore J_{KL} is strongly convex and has a unique minimizer.

We next compute the gradient. For a vector f, the derivative of F_m with respect to the source coordinate f_i is

$$\frac{\partial F_m}{\partial f_i}(f) = \sum_j \nu_j \frac{\exp((-C_{ij}^{(m)} + f_i)/\varepsilon)}{\sum_r \exp((-C_{rj}^{(m)} + f_r)/\varepsilon)}. \tag{B.67}$$

The right-hand side is the ith row mass of $\mathcal{C}_\nu(f; C^{(m)})$. At $f = X_m\omega$, the chain rule yields

$$\nabla_\omega J_{\mathrm{KL}}(\omega) = \sum_{m=1}^M X_m^\top \left(\widehat{P}_\omega^{0,(m)}\mathbf{1}_N - r^{(T,m)}\right) + \lambda\omega. \tag{B.68}$$

At the unique minimizer this gradient is zero, which proves the displayed identity. We take one more derivative of the row-mass expression. For $f = X_m\omega$,

$$\nabla_f^2 F_m(f) = \frac{1}{\varepsilon}\sum_j \nu_j \left(\mathrm{Diag}(\pi_j^{(m)}) - \pi_j^{(m)}(\pi_j^{(m)})^\top\right). \tag{B.69}$$

Each matrix inside the sum is a covariance matrix under the column conditional $\pi_j^{(m)}$, so it is positive semidefinite. The chain rule yields the displayed Hessian for J_{KL}, and the ridge term leaves the whole Hessian bounded below by λI_L. □

Proposition B.12. *For each fit example m, let y_m be the centered teacher source dual and let $P^{(T,m)} = \mathcal{C}_\nu(y_m; C^{(m)})$. For any coefficient vector ω, set $\widehat{f}_m = X_m\omega$ and $\widehat{P}_\omega^{0,(m)} = \mathcal{C}_\nu(\widehat{f}_m; C^{(m)})$. Assume the density condition in Proposition B.10 holds with the same constant β for each pair $(\widehat{P}_\omega^{0,(m)}, P^{(T,m)})$. Under this assumption, we have*

$$\frac{1}{M}\sum_{m=1}^M D_{\mathrm{KL}}\left(\widehat{P}_\omega^{0,(m)} \| P^{(T,m)}\right) \le \frac{\beta}{2\varepsilon^2 M}\sum_{m=1}^M \|X_m\omega - y_m\|_N^2. \tag{B.70}$$

Thus, conditional on the density assumption above, the ridge least-squares fit used by ASAP-LS minimizes an empirical quadratic upper bound on the average reverse teacher-plan KL over the chosen sliced-dual feature span. The minimizer of this surrogate need not coincide with the minimizer of the reverse plan KL itself; ASAP-KL in Proposition B.11 minimizes the forward KL for the one-sided key-normalized plan through the same source-dual parameterization.

Proof. Fix a calibration example m and apply the last bound in Proposition B.10 with $f = y_m$ and $\widehat{f} = X_m\omega$. Since $\|u\|_N^2 = N^{-1}\|u\|_2^2$, we have

$$D_{\mathrm{KL}}\left(\widehat{P}_\omega^{0,(m)} \| P^{(T,m)}\right) \le \frac{\beta}{2\varepsilon^2}\|X_m\omega - y_m\|_N^2. \tag{B.71}$$

We apply the same bound on every calibration example and average. □

A conditional random-feature bound.

Proposition B.10 shows how source-dual error controls the key-normalized Gibbs plan. We also need a modest statement about the sliced features themselves. We state the next result as an oracle bound. It assumes the centered teacher dual is already close to a square-integrable average of sliced potentials, then shows that independent slices recover that average at the usual Monte Carlo rate; it does not claim that arbitrary Sinkhorn teacher duals can be approximated by finitely many slices.

Proposition B.13. *Let $Z = (Q, K)$ be drawn from a calibration distribution, and let $y(Z) \in \mathbb{R}^N$ be the centered teacher source dual. Write $\|u\|_N^2 = N^{-1}\|u\|_2^2$. For each slice direction $\theta \sim \sigma$, let $\varphi_\theta(Z) \in \mathbb{R}^N$ be the centered sliced-potential feature. Assume that*

$$y(Z) = \mathbb{E}_{\theta\sim\sigma}[a(\theta)\varphi_\theta(Z)] + r(Z), \tag{B.72}$$

where

$$\mathbb{E}_{\theta\sim\sigma}a(\theta)^2 \le A^2, \qquad \|\varphi_\theta(Z)\|_N \le B, \qquad \mathbb{E}_Z\|r(Z)\|_N^2 \le \eta^2. \tag{B.73}$$

Draw $\theta_1, \dots, \theta_L$ independently from σ, and define the oracle coefficients $\omega_\ell^\star = a(\theta_\ell)/L$. If

$$R_\Theta(\omega) = \mathbb{E}_Z\left\|\sum_{\ell=1}^L \omega_\ell\varphi_{\theta_\ell}(Z) - y(Z)\right\|_N^2, \tag{B.74}$$

then

$$\mathbb{E}_\Theta R_\Theta(\omega^\star) \le \eta^2 + \frac{A^2B^2}{L}. \tag{B.75}$$

If, in addition, the density condition in Proposition B.10 holds with constant β for the key-normalized Gibbs plans induced by $y(Z)$ and by $\sum_\ell \omega_\ell^\star\varphi_{\theta_\ell}(Z)$, then

$$\mathbb{E}_{\Theta,Z} D_{\mathrm{KL}}\left(\widehat{P}_{\omega^\star}(Z) \,\|\, P^{(T)}(Z)\right) \le \frac{\beta}{2\varepsilon^2}\left(\eta^2 + \frac{A^2B^2}{L}\right). \tag{B.76}$$

Proof. Let

$$h(Z) = \mathbb{E}_{\theta\sim\sigma}[a(\theta)\varphi_\theta(Z)]\,, \qquad h_L(Z) = \frac{1}{L}\sum_{\ell=1}^{L} a(\theta_\ell)\varphi_{\theta_\ell}(Z). \tag{B.77}$$

Here h is the population sliced-potential average and h_L is its Monte Carlo approximation from the sampled slices. For a fixed input Z, write

$$\xi_\ell(Z) = a(\theta_\ell)\varphi_{\theta_\ell}(Z) - h(Z). \tag{B.78}$$

The vectors $\xi_1(Z), \dots, \xi_L(Z)$ are independent and have mean zero. The slice noise therefore has mean zero, so the mixed terms disappear in expectation and

$$\mathbb{E}_\Theta\|h_L(Z) - h(Z)\|_N^2 = \frac{1}{L}\mathbb{E}_{\theta\sim\sigma}\left\|a(\theta)\varphi_\theta(Z) - h(Z)\right\|_N^2. \tag{B.79}$$

The last term is bounded by the second moment:

$$\mathbb{E}_\Theta\|h_L(Z) - h(Z)\|_N^2 \le \frac{1}{L}\mathbb{E}_{\theta\sim\sigma}a(\theta)^2\|\varphi_\theta(Z)\|_N^2 \le \frac{A^2B^2}{L}. \tag{B.80}$$

Since $y(Z) = h(Z) + r(Z)$ and $\mathbb{E}_\Theta[h_L(Z) - h(Z)] = 0$, the cross term with $r(Z)$ also vanishes. We have

$$\mathbb{E}_\Theta\|h_L(Z) - y(Z)\|_N^2 = \mathbb{E}_\Theta\|h_L(Z) - h(Z)\|_N^2 + \|r(Z)\|_N^2. \tag{B.81}$$

Averaging over Z proves the displayed bound on $\mathbb{E}_\Theta R_\Theta(\omega^\star)$.

We now prove the KL bound. For each Z, apply Proposition B.10 with

$$\widehat{f}(Z) = h_L(Z), \qquad f(Z) = y(Z). \tag{B.82}$$

Since $\|\widehat{f}(Z) - f(Z)\|_N^2 = N^{-1}\|\widehat{f}(Z) - f(Z)\|_2^2$, we obtain

$$D_{\mathrm{KL}}\Big(\widehat{P}_{\omega^\star}(Z)\,\|\,P^{(T)}(Z)\Big) \le \frac{\beta}{2\varepsilon^2}\|\widehat{f}(Z) - f(Z)\|_N^2. \tag{B.83}$$

After averaging over Z and the slice draw, the source-dual error bound above proves the KL statement. □

Proposition B.14. *Let $Z_1, \dots, Z_M$ be calibration examples, let $X_m = X_\Theta(Z_m)$ be the sliced-feature matrix for example m, and let $y_m = y(Z_m)$ be the centered teacher source dual. Define*

$$R_M(\omega) = \frac{1}{M}\sum_{m=1}^{M}\|X_m\omega - y_m\|_N^2. \tag{B.84}$$

For $\lambda_{\mathrm{risk}} \ge 0$, let $\widehat{\omega}_{\mathrm{LS}}$ be any minimizer of

$$R_M(\omega) + \lambda_{\mathrm{risk}}\|\omega\|_2^2. \tag{B.85}$$

For every comparator $\omega^\circ \in \mathbb{R}^L$, we have

$$R_M(\widehat{\omega}_{\mathrm{LS}}) \le R_M(\omega^\circ) + \lambda_{\mathrm{risk}}\|\omega^\circ\|_2^2. \tag{B.86}$$

Now draw the slices as in Proposition B.13 and set $\omega_\ell^\circ = a(\theta_\ell)/L$. Under the assumptions of Proposition B.13,

$$\mathbb{E}_{\Theta, Z_{1:M}} R_M(\widehat{\omega}_{\mathrm{LS}}) \le \eta^2 + \frac{A^2B^2}{L} + \lambda_{\mathrm{risk}}\frac{A^2}{L}. \tag{B.87}$$

If the density condition in Proposition B.10 holds with constant β for the corresponding key-normalized Gibbs plans, then

$$\mathbb{E}_{\Theta, Z_{1:M}}\frac{1}{M}\sum_{m=1}^{M} D_{\mathrm{KL}}\Big(\widehat{P}^{(m)}_{\widehat{\omega}_{\mathrm{LS}}}\|P^{(T,m)}\Big) \le \frac{\beta}{2\varepsilon^2}\left(\eta^2 + \frac{A^2B^2}{L} + \lambda_{\mathrm{risk}}\frac{A^2}{L}\right). \tag{B.88}$$

The proposition controls the empirical calibration objective optimized by ASAP-LS. It is not a test-distribution generalization bound.

Proof. By optimality of $\widehat{\omega}_{\mathrm{LS}}$, its regularized empirical risk is no larger than that of any comparator. Therefore, for every ω° we have

$$R_M(\widehat{\omega}_{\mathrm{LS}}) + \lambda_{\mathrm{risk}}\|\widehat{\omega}_{\mathrm{LS}}\|_2^2 \le R_M(\omega^\circ) + \lambda_{\mathrm{risk}}\|\omega^\circ\|_2^2. \tag{B.89}$$

The left ridge term can only make the left side larger, so removing it keeps the inequality valid and proves the first claim.

Now choose the random-feature oracle comparator $\omega_\ell^\circ = a(\theta_\ell)/L$. The same calculation as in Proposition B.13 shows

$$\mathbb{E}_{\Theta, Z_{1:M}} R_M(\omega^\circ) \le \eta^2 + \frac{A^2B^2}{L}. \tag{B.90}$$

For the ridge term, we have

$$\mathbb{E}_\Theta\|\omega^\circ\|_2^2 = \frac{1}{L^2}\sum_{\ell=1}^{L}\mathbb{E}_{\theta_\ell}a(\theta_\ell)^2 \le \frac{A^2}{L}. \tag{B.91}$$

After averaging the first inequality, we obtain the empirical risk bound.

For the KL statement, we apply Proposition B.12 to $\omega = \widehat{\omega}_{\mathrm{LS}}$ and take expectation over the calibration examples and the slice draw. □

Proposition B.15. *In the unmasked square setting, let $\mu = N^{-1}\mathbf{1}_N$ and define $\rho(P) = \|P\mathbf{1}_N - \mu\|_1$. For two key-normalized probability plans P and $\widehat{P}$ with the same key marginal, using entrywise ℓ_1 for matrix gaps,*

$$\rho(\widehat{P}) \le \rho(P) + \|\widehat{P} - P\|_1 \le \rho(P) + \sqrt{2D_{\mathrm{KL}}(\widehat{P}\|P)}. \tag{B.92}$$

For attention-scale matrices $A = NP$ and $\widehat{A} = N\widehat{P}$,

$$\frac{1}{N}\sum_i\left|\sum_j \widehat{A}_{ij} - 1\right| = \rho(\widehat{P}). \tag{B.93}$$

Proof. Row summation is a contraction in ℓ_1. Combining this fact with the triangle inequality, we have

$$\|\widehat{P}\mathbf{1}_N - \mu\|_1 \le \|P\mathbf{1}_N - \mu\|_1 + \|(\widehat{P} - P)\mathbf{1}_N\|_1, \qquad \|(\widehat{P} - P)\mathbf{1}_N\|_1 \le \|\widehat{P} - P\|_1. \tag{B.94}$$

Pinsker's inequality proves the KL bound. Finally, since $A = NP$, we can write

$$\frac{1}{N}\sum_i\left|\sum_j \widehat{A}_{ij} - 1\right| = \frac{1}{N}\sum_i\left|N\sum_j \widehat{P}_{ij} - 1\right| = \sum_i\left|\sum_j \widehat{P}_{ij} - \frac{1}{N}\right| = \rho(\widehat{P}). \tag{B.95}$$

□

Proposition B.16. *Under the assumption of Proposition B.8, the multiplicative plan bound also controls one attention head. Let*

$$y_i = N\sum_j P_{f,ij}v_j, \qquad \widehat{y}_i = N\sum_j P_{\widehat{f},ij}v_j, \tag{B.96}$$

and assume $\|v_j\|_2 \le B$ for all j. Under this assumption, we have

$$\|\widehat{y}_i - y_i\|_2 \le NB\sum_j |P_{\widehat{f},ij} - P_{f,ij}| \le NB\big(e^{2\alpha/\varepsilon} - 1\big)\sum_j P_{f,ij}. \tag{B.97}$$

If the reference reconstructed plan has row mass $1/N$, this reduces to

$$\|\widehat{y}_i - y_i\|_2 \le B\big(e^{2\alpha/\varepsilon} - 1\big). \tag{B.98}$$

For finite-budget or masked conventions, the first bound applies with the actual row mass of the reference reconstructed plan.

Proof. Fix a query token i. The output error is the value-weighted gap between the two attention rows. By the triangle inequality and $\|v_j\|_2 \le B$, we have

$$\|\widehat{y}_i - y_i\|_2 \le NB \sum_j |P_{\widehat{f},ij} - P_{f,ij}|. \tag{B.99}$$

The multiplicative plan bound in Proposition B.8 also implies

$$|P_{\widehat{f},ij} - P_{f,ij}| \le \big(e^{2\alpha/\varepsilon} - 1\big) P_{f,ij}. \tag{B.100}$$

Using this pointwise bound inside the row sum proves the second displayed inequality. If the reference row mass is $1/N$, then $N \sum_j P_{f,ij} = 1$, which yields the simplified bound. □

The same output bound applies to the default ASAP plan after replacing Proposition B.8 by Proposition B.9.

Permutation equivariance.

Proposition B.17. *Consider the unmasked square setting with uniform marginals, and let* Π *be a permutation matrix. Assume slice sorting uses a deterministic tie rule that is consistent under token permutation. With this tie rule, the sliced feature matrix satisfies*

$$X_\Theta(\Pi Q, \Pi K) = \Pi X_\Theta(Q, K). \tag{B.101}$$

Consequently, ASAP satisfies

$$\widehat{f}_\omega(\Pi Q, \Pi K) = \Pi \widehat{f}_\omega(Q, K), \qquad \widehat{P}_\omega(\Pi Q, \Pi K) = \Pi \widehat{P}_\omega(Q, K)\Pi^\top, \tag{B.102}$$

and, for values permuted in the same way,

$$\mathrm{ASAP}(\Pi Q, \Pi K, \Pi V) = \Pi\ \mathrm{ASAP}(Q, K, V). \tag{B.103}$$

Proof. Fix a slice direction. Permuting Q and K permutes the projected source and target values, but the sorted order statistics are unchanged up to the same relabeling of token identities. Under the stated tie rule, the sliced source potential assigned back to token positions satisfies $\tilde{f}^{(\ell)}(\Pi Q, \Pi K) = \Pi \tilde{f}^{(\ell)}(Q, K)$ for every slice ℓ. Thus each sliced feature column permutes in the same way as the tokens, and $X_\Theta(\Pi Q, \Pi K) = \Pi X_\Theta(Q, K)$.

Centering commutes with permutation because $\Pi \mathbf{1}_N = \mathbf{1}_N$, so $\widehat{f}_\omega(\Pi Q, \Pi K) = \Pi \widehat{f}_\omega(Q, K)$. The cost matrix transforms as $C(\Pi Q, \Pi K) = \Pi C(Q, K)\Pi^\top$. Since the marginals are uniform in the square setting, the key-side and source-side c-transforms both commute with simultaneous token permutation. The one-sided ASAP-0 plan and the default two-sided ASAP plan therefore permute in the same way:

$$\widehat{P}_\omega(\Pi Q, \Pi K) = \Pi \widehat{P}_\omega(Q, K)\Pi^\top. \tag{B.104}$$

With $A = NP$, the value output is therefore

$$N\widehat{P}_\omega(\Pi Q, \Pi K)\Pi V = N\Pi \widehat{P}_\omega(Q, K)V = \Pi\ \mathrm{ASAP}(Q, K, V). \tag{B.105}$$

□

C Additional Related Work

Transport-based attention. Transport-based attention replaces row-softmax normalization with operators that impose additional structure on the attention plan. Sparse Sinkhorn Attention uses Sinkhorn balancing to learn block-wise relaxed permutations for sparse attention (Tay et al., 2020), while Sinkformers apply Sinkhorn scaling directly to the query–key score matrix. Other variants explore non-iterative, low-rank, and alternative parameterizations of doubly-stochastic attention, including LOTFormer, which uses a low-rank pivot coupling for linear-time inference (Shahbazi et al., 2025b). ASAP stays closest to the Sinkhorn line, keeping the dense trained Sinkhorn operator fixed and replacing only repeated normalization after training. Recent theory studies rank decay in Sinkhorn-style doubly-stochastic attention, while FlashSinkhorn accelerates entropic OT solvers at the kernel level without changing the transport operator (Ye et al., 2026).

Sliced optimal transport. Sliced OT uses the closed-form structure of one-dimensional OT as a computational primitive for high-dimensional measures (Nguyen, 2025). Recent work improves the choice of slices and extends the construction beyond Euclidean measures, including convolutional slices for images, adaptive slicing distributions, quasi-Monte Carlo slice sets, spherical and stereographic spherical sliced distances, Cartan-Hadamard geometry, and unbalanced transport (Nguyen & Ho, 2022; 2023; Nguyen et al., 2024; Bonet et al., 2023; Tran et al., 2024; Bonet et al., 2025a;b). Another line lifts one-dimensional slice solutions back into transport objects, including min-SWGG, Sliced OT Plans, generalized sliced Wasserstein plans, and min-STP (Mahey et al., 2023; Tanguy et al., 2025; Chapel et al., 2025; Liu et al., 2025a). Sliced-regularized OT instead uses a smoothed sliced OT plan as the reference coupling in a KL-regularized OT problem, with a Sinkhorn-style solver around an informative sliced prior (Nguyen, 2026). ASAP uses one-dimensional sliced potentials as token-level coordinates for calibrating a frozen Sinkhorn attention layer.

Amortized transport. Amortized OT methods learn reusable structure across transport problems. Meta OT predicts OT potentials or maps for repeated problems (Amos et al., 2023), and Sinkhorn-initialization methods use data-dependent dual initializers, including closed-form one-dimensional OT duals, to reduce solver time (Thornton & Cuturi, 2023). Wasserstein Wormhole learns embeddings whose Euclidean distances approximate Wasserstein distances (Haviv et al., 2024), and Nguyen et al. (2026) regress Wasserstein distances on sliced-Wasserstein features. Truong & Nguyen (2026) are closest in feature idea, using sliced Kantorovich potentials to amortize repeated OT solves. ASAP follows this sliced-potential construction inside a trained Transformer layer, where the object being compiled is the finite Sinkhorn normalizer with its learned score scale, mask, entropy, and update convention.

Other attention alternatives. A parallel literature improves attention without transport constraints. DiffTransformer subtracts two softmax operators to suppress attention noise (Ye et al., 2025). Sparse, low-rank, kernel, and Nyström approximations address the quadratic cost of vanilla attention (Kitaev et al., 2020; Wang et al., 2020; Choromanski et al., 2021; Xiong et al., 2021), while FlashAttention provides exact IO-aware softmax kernels (Dao et al., 2022).

D Transport Background

From row-softmax to doubly-stochastic attention. For an unmasked attention head, we write the queries, keys, and values as the rows q_i, k_j, v_j of $Q, K \in \mathbb{R}^{N\times d_h}$ and $V \in \mathbb{R}^{N\times d_v}$. The attention scores compare each query with every key through the scaled dot product

$$s_{ij} = \frac{q_i^\top k_j}{\sqrt{d_h}}. \tag{D.1}$$

Row-softmax turns each row of scores into a distribution over keys, so every query distributes one unit of attention. The total received by a key is unconstrained: several queries can place most of their weight on the same key. Doubly-stochastic attention also requires each key to receive one unit in total. Sinkhorn approaches these two constraints by alternately normalizing the rows and columns of the positive kernel $M^{(0)}_{ij} = \exp(s_{ij}/\varepsilon)$. We use $\varepsilon = 1$ unless stated otherwise. For a positive matrix M, the two normalization operations are

$$N_R(M)_{ij} = \frac{M_{ij}}{\sum_{r=1}^N M_{ir}}, \qquad N_C(M)_{ij} = \frac{M_{ij}}{\sum_{r=1}^N M_{rj}}. \tag{D.2}$$

We apply them in alternation to obtain the Sinkhorn iterations,

$$M^{(\ell+1)} = \begin{cases} N_R(M^{(\ell)}), & \ell \text{ even}, \\ N_C(M^{(\ell)}), & \ell \text{ odd}. \end{cases} \tag{D.3}$$

The first update is row-softmax applied to s/ε. The next update balances the total attention received by the keys, but also changes

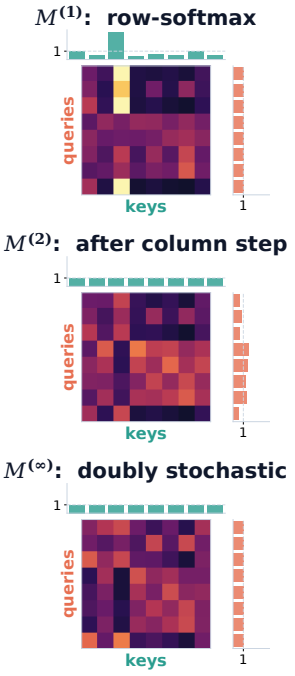


Figure D.1: Alternating row and column normalization.

how much each query sends. We therefore return to the rows, then to the columns, continuing this alternating adjustment toward a matrix whose row and column sums are both one. A trained Sinkformer uses a fixed number T of updates and returns the partially balanced attention matrix $A^{(T)} = M^{(T)}$.

Attention as transport. We can view attention as a transport plan by rescaling it to $P = A/N$, where P_{ij} is the mass sent from query i to key j. The row and column sums are its marginals: they describe how much mass leaves each query and arrives at each key. Under the doubly-stochastic constraint, each token has mass $1/N$, giving $\mu = \nu = \mathbf{1}_N/N$. More generally, for positive marginals μ and ν of total mass one, the set of plans with these marginals is

$$\mathcal{U}(\mu, \nu) = \left\{ P \in \mathbb{R}_+^{N \times N} \;\middle|\; P\mathbf{1}_N = \mu,\ P^\top \mathbf{1}_N = \nu \right\}. \tag{D.4}$$

To express the preference for nearby queries and keys, we use the scaled quadratic cost

$$C_{ij} = \frac{1}{2\sqrt{d_h}} \|q_i - k_j\|_2^2. \tag{D.5}$$

This cost equals $-s_{ij}$ plus terms that depend only on query i or key j. Every feasible plan sends the same total mass from each query and into each key, so these extra terms contribute the same value to the transport cost. The quadratic and negative-score costs therefore give the same balanced solution (Proposition B.1). We balance the preference for low-cost pairs against an entropy term that spreads mass among pairs. For $\varepsilon > 0$, the resulting entropic OT problem is

$$P^\star = \operatorname*{arg\,min}_{P \in \mathcal{U}(\mu,\nu)} \langle C, P \rangle + \varepsilon \sum_{i,j} P_{ij} \big(\log P_{ij} - 1 \big). \tag{D.6}$$

For positive marginals, the solution can be written using one scalar per query and one per key. We call these vectors the query potential $f^\star$ and key potential $g^\star$; they are the source and target dual potentials in OT terminology. The potentials describe the row and column rescalings needed to balance the kernel, and together they determine the optimal plan:

$$P^\star_{ij} = \exp\left(\frac{-C_{ij} + f^\star_i + g^\star_j}{\varepsilon} \right). \tag{D.7}$$

In this form, $e^{f_i^\star/\varepsilon}$ scales row i and $e^{g_j^\star/\varepsilon}$ scales column j of the kernel $\exp(-C_{ij}/\varepsilon)$. The potentials are expressed in cost units: an implementation that stores log scalings u, v represents them as $f^\star = \varepsilon u$ and $g^\star = \varepsilon v$. Thus two vectors describe the rescaling of the whole matrix, rather than a separate parameter for every query–key pair. We recover attention units through $A^\star = NP^\star$, whose row and column sums are one. The same exponential representation holds during the Sinkhorn computation, with the current potentials recording the rescalings accumulated so far.

Recovering the key potential. Once we know the query potential f, the column constraint determines the key potential. For each key j, the terms $\exp((-C_{ij} + f_i)/\varepsilon)$ give its unnormalized weights over queries. We choose the remaining factor $e^{g_j/\varepsilon}$ so that these weights sum to ν_j. The corresponding value of g_j is the key-side entropic c-transform,

$$T_K^\nu(f)_j = \varepsilon \log \nu_j - \varepsilon \log \sum_{i=1}^{N} \exp\left(\frac{-C_{ij} + f_i}{\varepsilon} \right). \tag{D.8}$$

The transform is column normalization expressed in potential coordinates: the scaling factor divides by the unnormalized column sum and multiplies by ν_j. The resulting plan therefore satisfies

$$\sum_i \exp\left(\frac{-C_{ij} + f_i + T_K^\nu(f)_j}{\varepsilon} \right) = \nu_j. \tag{D.9}$$

This relation lets us reconstruct a trained layer's attention from its query potential. For a teacher whose last update normalizes columns, we record $f^{(T)}$ immediately before that update and obtain $g^{(T)} = T_K^\nu(f^{(T)})$. These potentials describe the teacher at its chosen stopping point. We extract them from its own score kernel and normalization sequence, retaining its score scale, mask, and initialization; Appendix F describes their conversion to cost coordinates. ASAP-LS learns to predict

$f^{(T)}(Q, K)$, while ASAP-KL fits through the one-sided reconstruction using the teacher's plan as the reference,

$$P_{ij}^{(T)}(Q, K) = \exp\left(\frac{-C_{ij}(Q, K) + f_i^{(T)}(Q, K) + g_j^{(T)}(Q, K)}{\varepsilon}\right), \tag{D.10}$$

with $A^{(T)}(Q, K) = NP^{(T)}(Q, K)$. A constant added to the query potential is offset by the opposite shift in the key potential, leaving the plan unchanged. We therefore center $f^{(T)}$ before fitting, so the predictor learns its variation across queries. If the predicted potential agrees with $f^{(T)}$ up to a constant, ASAP-0 recovers the teacher's finite attention matrix. For other predictions, the reconstructed plan still has column sums ν, while its row sums and agreement with the teacher depend on the potential error. Default ASAP uses this one-sided plan as the starting point for the two-sided reconstruction in Equation 9.

E Additional Runtime Wall-Clock Analysis and Patch-Size Impact

This appendix supplements the main-paper runtime comparison with the full timing table, resource measurements, and implementation conventions.

E.1 FlashSinkhorn Discussion

ASAP and FlashSinkhorn reduce two parts of the Sinkhorn attention cost. FlashSinkhorn accelerates the same iterative Sinkhorn computation with IO-aware fused kernels, while ASAP changes the online operator after a teacher has been trained, replacing the iterative scaling loop of that fixed layer by sliced-feature extraction, one source-dual prediction, and the one-sided or two-sided entropic c-transform.

We compare two implementations. Table E.1 uses dense ASAP and the released fused iterative kernels, while Table E.3 uses the same streamed kernels for both ASAP and Sinkhorn. The former exposes the memory cost of storing the attention matrix; the latter measures the benefit of fewer updates when both methods avoid that storage. Appendix E.3 describes the streamed path.

For teacher fidelity, the relevant comparison is the frozen-layer tradeoff. Low-budget Sinkhorn spends fewer online updates, while ASAP spends one offline fit and replaces the iterative online scaling loop for the same fitted Sinkhorn parameterization.

E.2 Fused Sinkhorn Control

We use Table E.1 as a fused-kernel control on matched synthetic Q, K, V tensors. The public FlashSinkhorn kernels implement squared-Euclidean entropic OT, so this control uses unit-normalized queries and keys with the corresponding squared-Euclidean convention. The PyTorch rows reproduce a dense iterative reference implementation, the FlashSinkhorn rows use the released fused iterative kernels, and the ASAP rows report fitted one-sided and two-sided operators with a fixed analytic reconstruction. Because we use this synthetic setting only to compare implementations, the table reports latency and peak memory. Teacher-approximation errors are evaluated on trained-layer activations in the frozen replacement experiments.

Table E.1: Fused-Sinkhorn control under matched tensor shapes. Latency and peak memory are measured after CUDA warmup and explicit synchronization.

Method	Implementation	$N = 512$ ms ↓	$N = 2048$ ms ↓	$N = 4096$ ms ↓	$N = 4096$ GB ↓
Sinkhorn normalizer ($S = 3$)	PyTorch iterative	0.46	0.75	2.88	0.313
Sinkhorn teacher ($S = 20$)	PyTorch iterative	2.00	3.54	12.86	0.313
FlashSinkhorn ($S = 3$)	Fused iterative	0.37	**0.44**	**0.68**	**0.002**
FlashSinkhorn ($S = 20$)	Fused iterative	1.16	1.69	3.13	**0.002**
ASAP-LS-0 ($L = 8$)	PyTorch ASAP-0	0.33	0.71	1.80	0.313
ASAP-LS-0 ($L = 16$)	PyTorch ASAP-0	**0.32**	0.71	1.83	0.313
ASAP-LS-0 ($L = 32$)	PyTorch ASAP-0	0.33	0.72	1.86	0.313
ASAP-LS ($L = 32$)	PyTorch ASAP	0.41	0.88	2.44	0.313

At $N = 4096$, FlashSinkhorn cuts iterative memory from 0.313 GB to 0.002 GB and reduces the $S = 20$ pass from 12.86 ms to 3.13 ms. ASAP-0 runs in 1.80–1.86 ms for $L \in \{8, 16, 32\}$, and the default two-sided ASAP row runs in 2.44 ms. The fused $S = 3$ implementation remains faster and more memory efficient. The offline ASAP fit stays below 0.15 seconds across the measured lengths and slice counts.

The memory column reflects the implementations in this control. FlashSinkhorn streams the iterative computation, while the PyTorch ASAP reference materializes dense intermediate matrices. The shared-kernel comparison below uses streaming for both methods.

Although this control clarifies the implementations, it is not the teacher-fidelity evidence. In this normalized squared-Euclidean setting, low- and high-budget Sinkhorn are already close, so trained activations remain the relevant test for whether ASAP follows the trained finite operator.

E.3 Dense and Streamed Inference

The dense and streamed implementations evaluate the same predicted potentials and normalization steps. The dense path forms the score matrix, applies the updates, and multiplies the resulting attention by V. The streamed path uses the FlashSinkhorn transforms and its value-aggregation kernel, evaluating query–key interactions in tiles. With cost scale $1/(2\sqrt{d_h})$, we convert between score and squared-distance potentials using the shifts in Proposition B.1. We retain the finite teacher updates when extracting the target potentials; the shift changes their coordinates, not the teacher being approximated. The streamed implementation agrees with its floating-point reference to approximately 10^{-6} in the implementation check.

Table E.2 measures the dense path with six heads of width 64. All rows include score formation, the normalizer, and value mixing. These are attention-path timings, not complete encoder or classifier timings. We use one exclusively held RTX 6000 Ada GPU, 25 warmup iterations, and 100 timed iterations, reporting median latency. Three repeated measurements agree within 1.7%. This campaign is separate from the Blackwell measurements in Figure E.1.

Table E.2: Dense attention-path latency in milliseconds on RTX 6000 Ada. ASAP uses the $S = 10$ teacher. The speedup column compares default ASAP with that teacher.

N	Softmax	Sinkhorn $S = 3$	Sinkhorn $S = 10$	ASAP-0	ASAP	Speedup
197	0.033	0.187	0.444	0.289	0.355	1.25×
785	0.107	0.304	0.656	0.445	0.558	1.17×
2048	0.784	4.464	10.08	3.228	5.318	1.90×
8192	12.75	79.48	188.70	49.91	89.18	2.12×

For the streamed comparison in Table E.3, both the iterative baseline and ASAP use the same Triton kernels, entropy, and tensor shapes. We report median speedup over three runs, which agree within 2%. The kernels operate on two-dimensional inputs and loop over batch–head instances; this measurement covers inference only. Default ASAP requires three normalization passes, compared with ten for the teacher, while ASAP-0 requires one. At short lengths, the projections and sorting can absorb the savings from fewer passes. As the token count grows, the normalization work becomes dominant and the measured advantage increases.

Table E.3: Streamed inference using the same FlashSinkhorn kernels for both methods. Speedup is relative to fused Sinkhorn with $S = 10$; measurements use one RTX 6000 Ada GPU and summarize three timing runs. Peak memory is for default ASAP.

Tokens N	ASAP-0 speedup	ASAP speedup	ASAP peak GB
4096	1.85×	1.40×	0.017
16384	2.43×	1.78×	0.068
65536	3.02×	2.05×	0.246

Streaming avoids the $N \times N$ stored attention matrix. For fixed slice count, head width, and tile size, the working storage grows linearly with N, including the sliced features. Both methods still

perform quadratic query–key arithmetic. The peak-memory values in Table E.3 are measurements of the streamed ASAP path; the earlier control in Table E.1 instead compares dense ASAP with fused iterative Sinkhorn.

E.4 Dense Runtime Sweep

The sweep compares forward operators, not the cost required to obtain each trained model. Under this configuration, ESPFormer (Soft) runs out of memory at $N = 8000$, and ESPFormer (Hard) should be read as the post-training hard-sort inference operator reached after soft-sort training and temperature annealing.

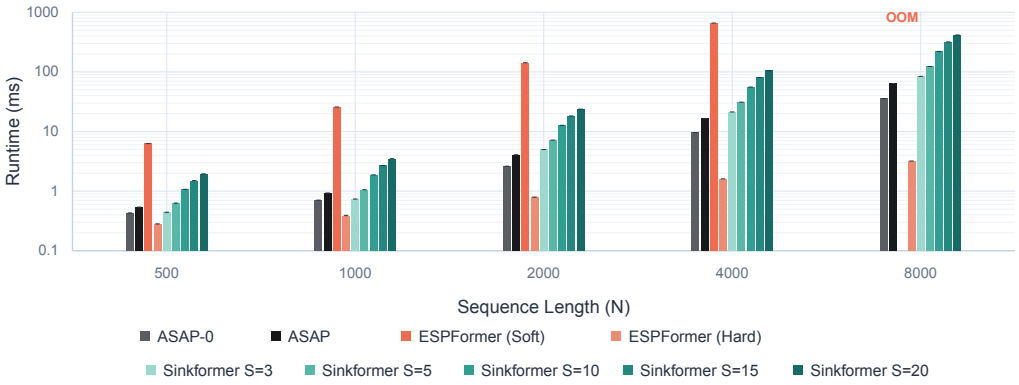


Figure E.1: Synthetic forward-latency sweep at batch size 1, with 32 slices for ASAP and ESPFormer. The vertical axis is logarithmic. Offline fitting is excluded. ESPFormer (Soft) runs out of memory at $N = 8000$; hard-sort rows measure its distinct inference operator. Table E.4 gives the ASAP and Sinkhorn values, and Table E.1 separately tests fused iterative kernels.

Table E.4: Forward latency in milliseconds for ASAP-0, ASAP, the Sinkhorn normalizer ($S = 3$), and the Sinkhorn teacher ($S = 20$) used for ASAP compilation in Figure E.1. Ratios are computed as baseline latency divided by ASAP-0 latency from the mean over 10 timed CUDA runs.

N	ASAP-LS-0	ASAP-LS	Sinkhorn normalizer ($S = 3$)	Speedup	Sinkhorn teacher ($S = 20$)	Speedup
500	0.42	0.53	0.43	1.02×	1.94	4.57×
1000	0.70	0.92	0.73	1.04×	3.49	4.96×
2000	2.61	4.06	4.97	1.90×	23.71	9.08×
4000	9.57	16.54	21.28	2.22×	104.72	10.95×
8000	35.70	63.51	84.03	2.35×	415.80	11.65×

E.5 ESPFormer Training Resources

ESPFormer trains through soft sorting before switching to hard-sort inference. The hard-sort inference operator is efficient, but the soft-sort training operator builds soft permutation tensors for queries and keys with shape $B \times H \times L \times N \times N$. We measure this operator because it is the training-time operator used before switching to hard sorting. The measurement follows the ESPFormer complexity discussion, where SoftSort and per-slice plan construction contribute $O(LN^2)$ terms, and the ESPFormer limitations note that training memory scales with the number of slices since each slice permutation matrix must be retained.

Figure E.3 plots the full IMDb encoder profile and a slice-count scaling profile at fixed $B = 4$, $N = 197$, and $H = 8$. Table E.5 reports the full IMDb resource sweep, and Figure E.2 shows the comparison normalized to Sinkformer $S = 3$. The measurement uses real IMDb token batches at sequence length 512 and effective batch size 32. ESPFormer (Soft) uses micro-batch 8 with four gradient-accumulation steps, so all train rows have the same effective batch size.

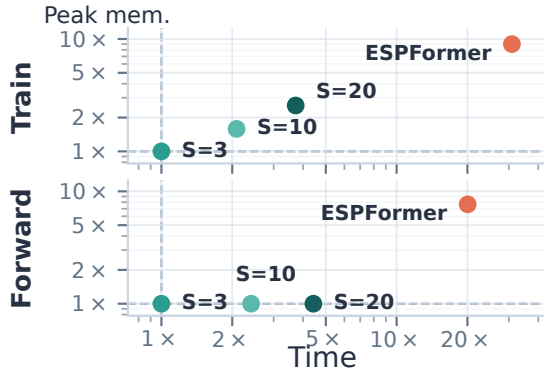


Figure E.2: IMDb train-step time and peak memory at sequence length 512, normalized to Sinkformer $S = 3$.

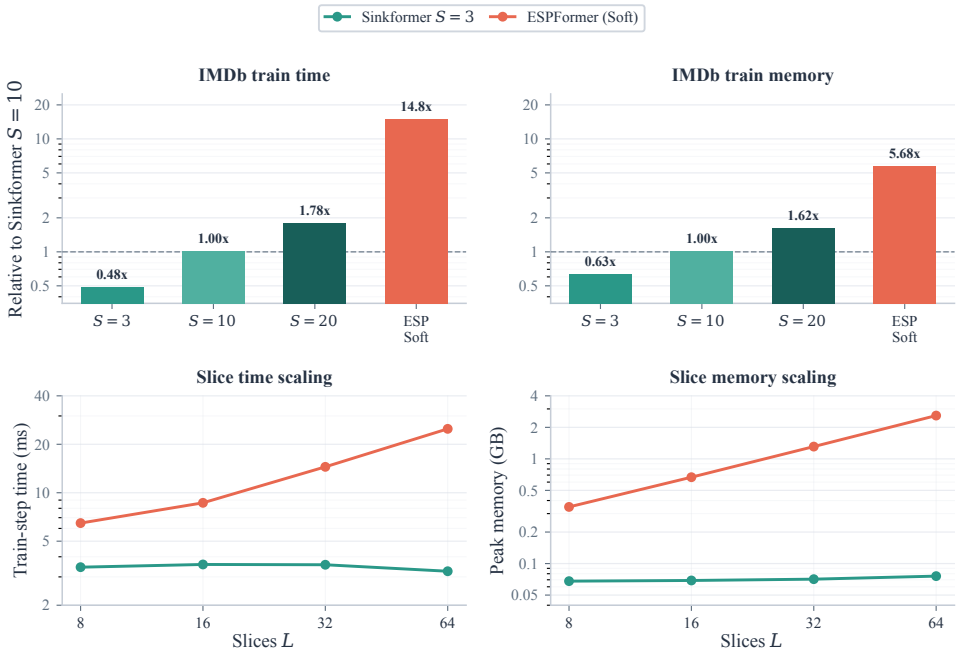


Figure E.3: ESPFormer training-resource profiles. Top panels use full IMDb encoder batches at sequence length 512, normalized to Sinkformer $S = 10$. Bottom panels show slice-count scaling at fixed $B = 4$, $N = 197$, and $H = 8$.

Table E.5: Full-encoder IMDb resource profile on RTX PRO 6000 Blackwell Server Edition GPUs. Train rows use effective batch size 32; forward rows report evaluation batches from the same sequence-length setting. Ratios are normalized to Sinkformer $S = 10$.

Method	Train ms/ex ↓	Train GB ↓	Train time vs. $S = 10$	Train mem. vs. $S = 10$	Forward ms/ex ↓	Forward GB ↓
Sinkformer ($S = 3$)	4.49	7.72	0.48	0.63	2.39	1.12
Sinkformer ($S = 10$)	9.37	12.24	1.00	1.00	5.75	1.12
Sinkformer ($S = 20$)	16.71	19.77	1.78	1.62	10.58	1.12
ESPFormer (Soft)	138.82	69.51	14.82	5.68	47.92	8.58

Table E.6 uses random Cats and Dogs ViT-shaped inputs to measure the attention-operator resource cost at the same batch size and image resolution as the downstream protocol. In this GPU measurement, ESPFormer (Soft) forward evaluation is $19.8\times$ slower and uses $43.1\times$ more peak GPU memory than Sinkformer ($S = 10$). ESPFormer (Hard) is slower still in this measurement because it constructs the soft-sort tensors before hardening. The ESPFormer (Soft) train step runs out of memory at batch size 64 on the same runtime, while Sinkformer train steps at $S \in \{3, 10, 20\}$ complete.

Table E.6: Full Cats and Dogs ViT-shape resource profile on random inputs of shape $64 \times 3 \times 224 \times 224$. Entries report mean wall-clock time after warmup and peak total allocated GPU memory. ESPFormer (Soft) uses the released soft-sort implementation, and ESPFormer (Hard) is a forward-only measurement because training uses the soft-sort row.

Method	Forward ms ↓	Forward GB ↓	Train-step ms ↓	Train-step GB ↓
Sinkformer ($S = 3$)	16.27	0.48	34.09	3.54
Sinkformer ($S = 10$)	38.84	0.48	79.88	6.90
Sinkformer ($S = 20$)	71.15	0.48	145.23	11.71
ESPFormer (Soft)	768.78	20.72	OOM	OOM
ESPFormer (Hard)	1603.83	30.82	–	–

The ESPFormer training-resource cells in Table G.2 come from the released soft-sort training log after dropping the first cold step. Because the run uses multiple devices, the table reports summed peak memory.

Table E.7 measures tensor-level train-step scaling with sequence length at batch size one and 64 slices. The soft-sort training operator retains dense per-slice sorting tensors, while Sinkhorn scaling keeps the online state closer to the attention matrix size.

Table E.7: Train-step scaling with sequence length on A100 GPUs, using batch size 1, 8 heads, and 64 slices. Each entry reports wall-clock milliseconds and peak allocated GPU memory in GB.

	Sinkformer ($S = 3$)		Sinkformer ($S = 10$)		ESPFormer (Soft)		ESPFormer (Hard)	
N	ms	GB	ms	GB	ms	GB	ms	GB
197	4.32	0.03	5.94	0.05	8.66	0.66	13.55	0.82
512	3.78	0.11	7.58	0.22	47.83	4.34	71.86	5.41
1000	5.09	0.35	12.59	0.80	229.42	16.49	292.00	20.59
2000	12.15	1.32	34.78	3.11	2216.72	65.85	2700.73	82.23

E.6 ESPFormer Soft-to-Hard Switch

ESPFormer uses soft sorting during training and reports hard sorting as an efficient inference operator after temperature annealing (Shahbazi et al., 2025a). A hard-sort gain is plausible when annealing has already moved the trained weights close to the hard-sorting regime. In that case, the soft operator behaves like a noisy relaxation, and hard permutations can sharpen the intended sliced transport plan. On an easier binary task such as Cats and Dogs, the classifier can also tolerate a larger logit change if the decision boundary remains stable.

The same switch can substantially change the evaluated operator. Hard sorting is a discontinuous limit of the soft relaxation and produces much lower-entropy attention. To quantify the effect, we evaluate the same CIFAR-100 ESPFormer checkpoint under three operators: the soft training relaxation, the released SoftSort hard flag, and the exact argsort operator that sorts each query and key slice, matches equal ranks, and averages the slice plans. The released hard flag hardens the soft relaxation by top-one selection, but it is not the exact permutation operator. The exact argsort operator is the relevant hard-sort reference for marginal balance.

Table E.8 compares the two hard-sort variants. ESPFormer (Exact Hard) restores the row and column marginals and is fastest in this run. Its task behavior still does not match ESPFormer (Soft), with test accuracy dropping from 41.89 to 36.75 and predictions changing on 34.65% of test examples. ESPFormer (Released Hard) is less balanced and lower in accuracy than ESPFormer (Exact Hard).

Table E.8: CIFAR-100 soft-to-hard comparison for the same epoch-50 ESPFormer checkpoint. Test accuracy and prediction change are measured on the test set, with prediction change relative to ESPFormer (Soft). Row and column errors are mean absolute deviations from one, averaged over recorded attention layers and batches. Throughput is measured on the validation split with batch size 256.

Method	Test acc. (%)	Pred. change (%)	Row err.	Col err.	Entropy	Val ex./s
ESPFormer (Soft)	41.89	0.00	0.059	0.060	2.858	948
ESPFormer (Released Hard)	35.31	41.02	0.148	0.115	0.176	451
ESPFormer (Exact Hard)	36.75	34.65	5.9×10^{-4}	5.9×10^{-4}	0.186	6572

We do not read this as a contradiction of the published Cats and Dogs hard-sort row; instead, it shows that hard-sort ESP inference is another operator whose task effect should be measured, not a cheaper evaluation of the same trained soft-sort operator. ASAP avoids that soft-to-hard operator switch by training the Sinkhorn operator directly and compiling the finite Sinkhorn duals after teacher training.

E.7 Patch-Size Impact under Sinkhorn Layer Replacement

The shallow MNIST experiment follows the one-layer ViT protocol used by Sinkformers and ESPFormer, where each attention mechanism is trained end to end across patch sizes. We use the same setting for replacement by training a Sinkhorn teacher ($S = 5$), freezing its learned projections and classifier, and replacing only the attention normalization at evaluation time. We report the stable, nontrivial replacement settings, patch sizes $2, 4, 7, 14$. ASAP-0 is the one-sided key-side ablation, and ASAP is the row-ending two-sided transform for this $S = 5$ teacher.

Table E.9: Patch-size impact under Sinkhorn layer replacement. Each entry is accuracy / teacher agreement in percent, reported as mean $\pm$ standard deviation over three runs. Teacher agreement is prediction agreement with the frozen Sinkhorn teacher ($S = 5$).

Method	Patch 2	Patch 4	Patch 7	Patch 14
Sinkhorn teacher ($S = 5$)	96.87 ± 0.18 / 100.00	96.89 ± 0.52 / 100.00	96.27 ± 0.30 / 100.00	96.32 ± 0.12 / 100.00
Softmax replacement	24.20 ± 12.37 / 24.34 ± 12.53	10.77 ± 1.60 / 10.76 ± 1.73	13.89 ± 1.36 / 13.88 ± 1.44	19.06 ± 5.75 / 19.07 ± 5.72
Sinkhorn normalizer ($S = 3$)	89.51 ± 2.66 / 90.22 ± 2.56	90.75 ± 4.45 / 91.77 ± 4.78	76.86 ± 11.26 / 77.54 ± 11.50	92.94 ± 3.59 / 94.31 ± 3.86
ASAP-0	94.74 ± 0.06 / 95.70 ± 0.12	94.61 ± 1.40 / 96.18 ± 1.31	88.76 ± 9.06 / 90.12 ± 9.22	95.14 ± 1.34 / 97.45 ± 1.57
ASAP	96.03 ± 0.18 / 97.28 ± 0.10	96.19 ± 0.82 / 97.96 ± 0.80	95.18 ± 1.30 / 97.27 ± 1.50	96.29 ± 0.13 / 99.25 ± 0.29

With one attention layer and no feed-forward block, this is a sensitive replacement test with little redundancy for normalization mismatch. On patch size 2, ASAP-0 substantially improves over the Sinkhorn normalizer ($S = 3$) replacement, reaching 94.74% accuracy and 95.70% teacher agreement. ASAP recovers most teacher behavior, reaching 96.03% accuracy and 97.28% agreement; after five post-replacement epochs on patch size 2, ASAP reaches $96.87 \pm 0.15\%$ accuracy, matching the Sinkhorn teacher continuation within run-to-run variation.

Across patch sizes $4, 7, 14$, ASAP stays close to the teacher, with mean accuracy drops of 0.71, 1.10, and 0.02 percentage points, respectively, while maintaining substantially higher teacher agreement than the Sinkhorn normalizer ($S = 3$) replacement. This sensitive setting shows the value of ASAP when a low-redundancy architecture needs tighter normalization fidelity.

F Implementation Details

Algorithm F.1 gives the fit and reconstruction for one predictor. We can pool heads within a layer for a shared fit or run the same procedure separately for each head. TeacherSource returns the query potential used before the teacher's last normalization. For a column-ending teacher, ASAP evaluates the key, query, and key transforms in sequence. The algorithm uses $\hat{A}$ directly for value mixing; the ImageNet classification comparison also evaluates final row normalization (Appendix I).

This section fixes the implementation choices that couple the frozen Sinkhorn teacher to the compiled ASAP layer.

Algorithm F.1 ASAP offline fit and online layer

Offline sliced-dual projection

Require: Fit set $\mathcal{D}$, slices Θ, teacher budget T, entropy ε, ridge λ, objective

$G \leftarrow \lambda I_L, b \leftarrow 0$
for $m = 1, \dots, M$ **do**
  $X_m \leftarrow X_\Theta(Q^{(m)}, K^{(m)})$
  $C^{(m)} \leftarrow \textsc{Cost}(Q^{(m)}, K^{(m)})$
  $f^{(T,m)} \leftarrow \textsc{TeacherSource}_T(Q^{(m)}, K^{(m)})$
  $\tilde{f}^{(T,m)} \leftarrow \Pi_0(f^{(T,m)})$
  if objective is LS **then**
    $G \leftarrow G + X_m^\top X_m$
    $b \leftarrow b + X_m^\top \tilde{f}^{(T,m)}$
  else
    $P^{(T,m)} \leftarrow \mathcal{C}_\nu(\tilde{f}^{(T,m)}; C^{(m)})$
    store $(X_m, C^{(m)}, P^{(T,m)})$ for Eq. B.64
  end if
end for
if objective is LS **then**
  $\omega \leftarrow G^{-1} b$
else
  $\omega \leftarrow$ minimizer of Eq. B.64
end if
return ω

Online compiled attention

Require: Evaluation pair (Q, K), values V, slices Θ, coefficient vector ω, teacher last step $s \in \{\text{row}, \text{column}\}$

$X \leftarrow X_\Theta(Q, K)$
$C \leftarrow \textsc{Cost}(Q, K)$
$\hat{f} \leftarrow \Pi_0(X\omega)$
if ASAP-0 **then**
  $\hat{g}^0 \leftarrow T_K^\nu(\hat{f})$
  $\hat{P}_{ij} \leftarrow \exp\left(\frac{-C_{ij} + \hat{f}_i + \hat{g}_j^0}{\varepsilon}\right)$
else if ASAP **then**
  $\hat{g}^0 \leftarrow T_K^\nu(\hat{f})$
  $\hat{f}^+ \leftarrow T_Q^\mu(\hat{g}^0)$
  if $s =$ column **then**
    $\hat{g}^+ \leftarrow T_K^\nu(\hat{f}^+)$
    $\hat{P}_{ij} \leftarrow \exp\left(\frac{-C_{ij} + \hat{f}_i^+ + \hat{g}_j^+}{\varepsilon}\right)$
  else
    $\hat{P}_{ij} \leftarrow \exp\left(\frac{-C_{ij} + \hat{f}_i^+ + \hat{g}_j^0}{\varepsilon}\right)$
  end if
end if
$\hat{A} \leftarrow N\hat{P}$
return $\hat{A}V$

F.1 Teacher Attention and Offline Fit

The normalized risk statements in Appendix B use λ_{risk}. Under that normalization, the implementation ridge in Equation 5 is $\lambda = MN\lambda_{\text{risk}}$.

ASAP is fit to the finite-budget teacher used by the trained model, not an exact-convergence idealization. For each task, we use the teacher's entropy, iteration budget, mask, and update convention when extracting duals or plans. ASAP-LS centers the source log-dual and solves the sliced-potential regressor by closed-form ridge fitting. ASAP-KL uses the same sliced features and source-dual parameterization, but fits the coefficients through the teacher plan after the key-side c-transform.

The experiments in this paper use a short post-training fitting pass after the teacher is frozen. The same LS normal equations may also be accumulated during a late training phase, but if the teacher parameters change during accumulation the fit corresponds to the sequence of observed teacher states, not exactly to the final frozen teacher.

We use the score-to-cost conversion because the teacher is implemented in score-kernel coordinates, while sliced potentials are quadratic-cost features. When the teacher backbone is defined from scaled dot-product scores, teacher extraction and inference-time reconstruction use that same finite score-kernel convention. The score-kernel source dual is first shifted into quadratic-cost coordinates by adding $\rho_i(Q) = \frac{1}{2\sqrt{d_h}}\|q_i\|_2^2$ before fitting. At inference time the predicted quadratic-cost source dual is shifted back by subtracting $\rho_i(Q)$ before the score-kernel c-transforms. The conversion is a finite-coordinate conversion for the implemented score-kernel teacher. It is not a claim that finite score-kernel Sinkhorn and finite quadratic-cost Sinkhorn from zero initialization have identical iterates. In particular, the sentiment experiments follow the released Sinkformer text score-kernel implementation while applying this dual-coordinate conversion.

Fit examples are drawn from the same unlabeled in-distribution training pipeline as the teacher and are processed with the same tokenizer or image preprocessing. Unless noted otherwise, the fit is performed per attention layer, with token positions and head instances pooled within that layer, and the same slice set Θ is reused across all fit and evaluation examples for a given compiled layer. In plug-and-play evaluation, the compiled ASAP layer is used after fitting; when a post-replacement refresh updates upstream query and key projections, gradients through hard sorting are treated as the

implementation's almost-everywhere sort gradients unless a given experiment explicitly freezes those features.

Masked text attention. For padded text batches, let $J(x) \subseteq \{1, \dots, N\}$ denote the active key positions for an input x. The released sentiment implementation masks padded key columns but keeps all N query positions in the sequence. We match this convention for teacher extraction, ASAP fitting, and ASAP reconstruction. In coupling units, the masked key target is

$$\nu_j^J = \begin{cases} 1/N, & j \in J(x), \\ 0, & j \notin J(x). \end{cases} \tag{F.1}$$

For $j \in J(x)$, the key-side c-transform is

$$\hat{g}_{\omega,j} = \varepsilon \log(1/N) - \varepsilon \log \sum_{i=1}^{N} \exp\left(\frac{-C_{ij} + \hat{f}_{\omega,i}}{\varepsilon} \right), \tag{F.2}$$

and for $j \notin J(x)$ we set $\hat{g}_{\omega,j} = -\infty$ and $\hat{P}_{\omega,ij} = 0$. With the attention-scale matrix $\hat{A}_\omega = N\hat{P}_\omega$, each active key column has target sum one and padded columns have target sum zero.

When $|J(x)| < N$, the masked object has total mass $|J(x)|/N$, so the text experiments compile the finite key-masked Sinkformer convention, not the square balanced transport ideal used in the main text. The balanced positive-marginal propositions in Appendix B apply to the unmasked square setting. For masked sliced features, sorting is applied on active token positions only, and padded positions do not enter the one-dimensional key support or the regression loss. Row errors in the sentiment experiments are reported in their own table and reflect the active-key mask and final normalization convention as well as source-dual approximation error. Column errors are averaged only over active key columns.

Proposition F.1. *Let $J \subseteq \{1, \dots, N\}$ be the active key set and define the masked key target*

$$\nu_j^J = \begin{cases} 1/N, & j \in J, \\ 0, & j \notin J. \end{cases} \tag{F.3}$$

For any source dual $f \in \mathbb{R}^N$, define, for $j \in J$,

$$g_j^J = \varepsilon \log(1/N) - \varepsilon \log \sum_{i=1}^{N} \exp\left(\frac{-C_{ij} + f_i}{\varepsilon} \right), \tag{F.4}$$

and set $g_j^J = -\infty$ for $j \notin J$. With the convention $\exp(-\infty) = 0$, let

$$P_{ij}^J = \exp\left(\frac{-C_{ij} + f_i + g_j^J}{\varepsilon} \right). \tag{F.5}$$

With this convention, we have

$$\sum_i P_{ij}^J = 1/N \quad \text{for } j \in J, \qquad P_{ij}^J = 0 \quad \text{for } j \notin J. \tag{F.6}$$

The total mass of P^J is $|J|/N$.

Proof. For an active key $j \in J$, exponentiating the definition of g_j^J yields

$$\exp\left(\frac{g_j^J}{\varepsilon} \right) = \frac{1/N}{\sum_i \exp\left(\frac{-C_{ij} + f_i}{\varepsilon} \right)}. \tag{F.7}$$

For an active key, the same cancellation as in the unmasked case leaves exactly the active-key mass:

$$\sum_i P_{ij}^J = 1/N. \tag{F.8}$$

For an inactive key $j \notin J$, the convention $\exp(-\infty) = 0$ forces $P_{ij}^J = 0$ for all i. Summing the active column masses, the total mass is $|J|/N$. □

F.2 Online and Offline Cost

For one dense forward pass, ASAP requires $O(N^2 d_h)$ work to form the dense score or cost matrix, $O(LNd_h)$ for slice projections, $O(LN \log N)$ for slice-wise sorting and potential extraction, $O(NL)$ for dual prediction, $O(N^2)$ work per entropic c-transform call, and $O(N^2 d_v)$ for multiplying the dense attention matrix by the value matrix. ASAP-0 uses one c-transform call. Column-ending ASAP uses the three calls in Equation 9; row-ending ASAP uses two calls. In implementation, the key-side transform followed by multiplication by N is evaluated in its algebraically equivalent column-wise softmax form. In the unmasked square ASAP-0 case,

$$\hat{A}_{\omega,ij} = \frac{\exp\Big((-C_{ij} + \hat{f}_{\omega,i})/\varepsilon\Big)}{\sum_{i'=1}^{N} \exp\Big((-C_{i'j} + \hat{f}_{\omega,i'})/\varepsilon\Big)}. \tag{F.9}$$

The dense online cost is $O(N^2 d_h + LNd_h + LN \log N + NL + RN^2 + N^2 d_v)$ with $R = 1$ for ASAP-0, $R = 3$ for column-ending ASAP, and $R = 2$ for row-ending ASAP, compared with $O(N^2 d_h + SN^2 + N^2 d_v)$ for S Sinkhorn scaling steps after score formation. For M fit examples, the offline work includes teacher extraction, typically $O(MTN^2)$ for T teacher sweeps after costs are formed, sliced-feature computation, and coefficient fitting. ASAP-LS adds $O(MNL^2)$ statistic accumulation and an $O(L^3)$ solve per fitted layer. ASAP-KL adds a low-dimensional convex optimization over the same L coefficients, where each objective or gradient pass evaluates the key-normalized Gibbs plan on the calibration batch. These offline terms are not part of the online forward latency reported in the runtime tables.

The offline fit pays off after repeated evaluations of the same frozen layer. In the frozen IMDb sweep, ASAP-LS with $L = 32$ and $M = 4096$ uses 102.0 seconds of fit time. The default two-sided row saves $63.02 - 11.79 = 51.23$ ms per layer-batch relative to the $S = 20$ teacher, for a break-even point of approximately 2.0×10^3 layer-batch evaluations. Relative to Sinkformer ($S = 3$), the saving is $14.14 - 11.79 = 2.35$ ms, for approximately 4.3×10^4 layer-batch evaluations. The faster ASAP-LS-0 ablation uses the same fit time and saves $63.02 - 7.74 = 55.28$ ms against the $S = 20$ teacher and $14.14 - 7.74 = 6.40$ ms against Sinkformer ($S = 3$), for break-even points of approximately 1.8×10^3 and 1.6×10^4 layer-batch evaluations, respectively. These counts use forward latency only and quantify the repeated-inference regime targeted by post-training fitting.

F.3 Timing and Evaluation Protocol

The runtime sweep in Figure E.1 was executed on NVIDIA RTX PRO 6000 Blackwell Server Edition GPU hardware with PyTorch 2.10.0+cu128. We use batch size 1, sequence lengths $N \in \{500, 1000, 2000, 4000, 8000\}$, width $d = 1024$, 8 attention heads, and head dimension 64. Sinkhorn normalizer baselines are benchmarked at iteration budgets $S \in \{3, 5, 10, 15, 20\}$. ESPFormer uses soft-sort temperature $t = 10.0$ and inverse-temperature parameter $\tau = 10^{-3}$. ASAP uses $L = 32$ slices, ridge $\lambda = 10^{-3}$, 8 fit examples, fit batch size 2, and a Sinkhorn teacher ($S = 20$). All timings use 3 warmup runs, 10 timed runs, and explicit CUDA synchronization around each measurement. Figure E.1 and Table E.4 report forward latency only; the one-time ASAP fit is reported when fit cost is part of the comparison.

G Experiment Details

Table G.1 fixes the main experimental settings before the task-specific details.

Each paragraph below lists the architecture, optimizer, schedule, and ASAP fit used by the corresponding main-paper result. When Sinkformer and ESPFormer do not use the same shared settings, we follow the ESPFormer reproduction for the main comparison and note the mismatch locally.

G.1 Vision Results and Validation Curves

Tables G.2 and G.3 give the Cats and Dogs and CIFAR-100 comparisons. The Cats and Dogs table includes training memory and compilation time; the CIFAR-100 panel shows validation loss for the independently trained baselines.

Table G.1: Settings for the text and smaller vision benchmarks. The shared vision comparisons follow the ESPFormer reproduction protocol; the paragraphs below give task-specific details. ImageNet settings are in Appendix I.

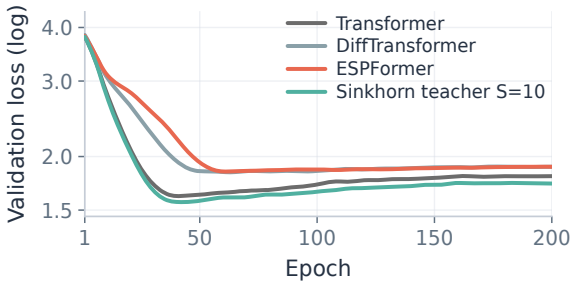
Benchmark	Training and evaluation setting	ASAP-specific setup
IMDb replacement	New matched benchmark with an IMDb encoder at sequence length 512. Resource profile uses effective batch size 32 and three trained $S = 20$ teachers.	Compile each frozen $S = 20$ teacher and report resource use, layer-batch latency, teacher error, marginal error, and frozen accuracy.
Cats and Dogs	ESPFormer reproduction of the Sinkformer setup with a ViT using embed dim 128, depth 6, 8 heads, patch size 16, Adam, batch size 64, and 300 epochs.	Compile after teacher training and report the full-data setting. High-S rows are controlled extensions.
Text classification	Controlled topic and sentiment benchmark set with a depth-6 encoder, batch size 32, 15 epochs, learning rate 10^{-4}, and sequence length 128.	Compile from the $S = 10$ teacher using 4096 unlabeled training inputs. Rows share the same checkpoint rule.

Table G.2: Cats and Dogs full-data ViT results. ESPFormer rows marked † are published results. ASAP rows compile three Sinkhorn teachers ($S = 10$), and resource cells use released ESPFormer implementations.

Method	Acc. (%) ↑	Train peak (GB) ↓	Compile (s) ↓
Transformer	$78.49 \pm 0.09^{\dagger}$	1.64	–
DiffTransformer	$78.85 \pm 0.11^{\dagger}$	4.22	–
Sinkformer ($S = 3$)	$79.12 \pm 0.17^{\dagger}$	3.54	–
ESPFormer (Soft, init)	$79.47 \pm 0.12^{\dagger}$	170.45	–
ESPFormer (Soft, sharp)	$80.61 \pm 0.11^{\dagger}$	170.45	–
ESPFormer (Hard)	$81.23 \pm 0.11^{\dagger}$	170.45	–
Sinkhorn teacher ($S = 10$)	81.09 ± 0.61	6.90	–
ASAP-LS-0 ($L = 32$)	80.15 ± 0.69	6.90	24.38 ± 0.16
ASAP-KL-0 ($L = 32$)	79.77 ± 0.90	6.90	101.19 ± 0.76
ASAP-LS ($L = 64$)	80.44 ± 0.39	6.90	25.02 ± 0.17

Table G.3: CIFAR-100 patch-size-4 control. The left panel reports final accuracy for trained baselines and ASAP replacements. The right panel reports validation loss for trained baselines only.

Method	Acc. (%) ↑
Transformer	62.73 ± 0.01
DiffTransformer	60.03 ± 0.59
ESPFormer	58.46 ± 0.42
Teacher ($S = 10$)	$\mathbf{64.95 \pm 0.06}$
ASAP-LS-0 ($L = 32$)	63.64 ± 0.42
ASAP-KL-0 ($L = 32$)	63.58 ± 0.66
ASAP-LS ($L = 32$)	63.46 ± 0.36
ASAP-KL ($L = 32$)	63.34 ± 0.44



G.2 Operator-Level Benchmarks

The main frozen-layer replacement benchmark uses trained IMDb encoders with Sinkhorn attention. For each run, we train a Sinkhorn teacher at budget $S = 20$, freeze the model, and replace only the attention normalizer at evaluation time. The Sinkhorn normalizer ($S = 3$) row reuses the same trained query, key, value, and output projections but runs only three Sinkhorn updates. ASAP rows reuse the same projections and substitute the fitted operator with the one-sided or two-sided c-transform. Table 1 reports layer-batch runtime, teacher-relative output RMSE, attention relative ℓ_2 error, and

final frozen-classifier accuracy over three trained teachers. Table H.1 reports the corresponding row and column marginal errors, and Table H.2 reports the ASAP-LS slice-count sweep on the same held-out batch-layer cases.

The synthetic runtime sweep in Figure E.1 uses matched dense self-attention inputs at sequence lengths $N \in \{500, 1000, 2000, 4000, 8000\}$, batch size 1, model width $d = 1024$, 8 attention heads, and head dimension 64. Runtime is averaged over 10 timed runs after 3 warmup runs with explicit CUDA synchronization. ESPFormer (Soft) and ESPFormer (Hard) use 32 slices under the shared synthetic width setting. These synthetic rows are used for forward-latency context, not as teacher-agreement evidence for the trained IMDb layer, because ESPFormer constructs another expected sliced transport plan.

G.3 Cats and Dogs

For Cats and Dogs, we use the ESPFormer reproduction of the Sinkformer ViT setup. Images are resized to 224×224 and processed by a ViT with embedding dimension 128, MLP dimension 128, depth 6, 8 attention heads, and patch size 16. The main comparison uses the full training set. Training uses Adam, batch size 64, 300 epochs, and an initial learning rate of 3×10^{-5}, reduced by a factor of 10 after 250 epochs. Normalization and data augmentation follow the Sinkformer protocol. The published baseline accuracies are the three-run results from ESPFormer. We train the Sinkhorn teachers with this backbone and optimization schedule, then fit ASAP once before evaluation.

The reported teacher and replacement rows use the archived best-validation checkpoints for seeds 1 and 2 and the final-epoch checkpoint for seed 3. Each replacement is evaluated against the checkpoint from which it was calibrated. For Table 4, we time a full model forward pass on the first validation batch of 64 GPU-resident images, after five warmup passes. We average 20 timed passes per model, with CUDA synchronization around each pass, then report mean and standard deviation across the three runs. Data loading and host-to-device transfer occur before timing. The one-sided Cats and Dogs variant applies a final row normalization before value mixing; the corrected variant uses the column-ending reconstruction without that additional step.

G.4 Text Classification

The text-classification experiments use the same encoder-plus-pooling backbone family as IMDb, with depth 6, batch size 32, Adam, 15 epochs, and hidden width 256 with 8 attention heads. We use maximum sequence length 128 on the standard DBpedia-14, AG News, and Yelp Review Full splits; the tasks have 14, 4, and 5 classes, respectively. The tokenizer assets are shared with the self-contained sentiment notebooks, but all models are trained from scratch on their target dataset with no transfer from IMDb. Sinkhorn teachers use the score-kernel Sinkhorn convention with active-key masking for padded text. The main table reports mean and standard deviation over three runs under a shared best-evaluation checkpoint rule. ASAP is fit from the Sinkhorn teacher ($S = 10$) on the first 4096 training examples, using only inputs and discarding labels.

G.5 Shallow MNIST Layer Replacement Protocol

The shallow MNIST layer-replacement experiment follows the one-layer ViT setting used in Sinkformer and ESPFormer to focus on the attention mechanism instead of deeper architectural choices. Images are processed by a single-layer self-attention model with one head and no feed-forward block. Training uses Adam, batch size 100, and 45 epochs. As in the reference papers, the Transformer-style baselines use an initial learning rate of 10^{-3}, whereas the Sinkhorn-style variants use 2×10^{-3}. In both cases the learning rate is reduced by a factor of 10 after epochs 35 and 41.

ASAP uses 256 sliced-potential features, a residual log-sum-exp dual fit, and 8192 unlabeled calibration examples. The $S = 5$ teacher ends with a row normalization, so ASAP uses the row-ending two-sided transform. ASAP-0 is the fitted one-sided key-side ablation.

The reported refresh updates the ASAP coefficients, token bias, output projection, layer normalization, and classifier head for five supervised epochs at learning rate 2×10^{-4}. The Sinkformer ($S = 5$) continuation row uses the same number of extra epochs as a control. Table E.9 reports patch-size replacement results over three runs.

H Additional Ablations

H.1 Further Results for the Frozen-Layer Replacement

The frozen-layer results compare latency with fidelity. The $S = 3$ and $S = 5$ normalizers are faster than the $S = 20$ teacher because they stop the same scaling loop earlier, but they execute another finite operator. ASAP-0 shows the raw learned-dual projection speed by computing only the key-side transform. The default ASAP operator applies the two-sided transform, spends a small fixed number of log-sum-exp passes, and stays much closer to the teacher without rerunning the full scaling loop.

These results address the two questions that guide the paper, whether Sinkhorn attention is useful during training and whether ASAP preserves the trained operator after replacement. Sinkhorn attention improves the teacher rows most consistently on CIFAR-100 and text, whereas ASAP measures how much of that learned computation remains after we replace most of the repeated scaling loop. Sequence classification transfers tightly with the default two-sided transform, while CIFAR-100 has a larger reported accuracy gap.

The main operator table measures teacher agreement. The marginal table below checks which side of the Sinkhorn constraint each replacement preserves. For an attention matrix A, let $r_i(A) = \sum_j A_{ij}$ and $c_j(A) = \sum_i A_{ij}$. On padded IMDb batches, $\mathcal{J}$ denotes the active non-padded key columns. We report $\Delta_{\mathrm{row}\to 1} = N^{-1} \sum_i |r_i(A) - 1|$ and $\Delta_{\mathrm{col}\to 1} = |\mathcal{J}|^{-1} \sum_{j\in\mathcal{J}} |c_j(A) - 1|$. These errors depend on the last row or column normalization under key padding. Sinkhorn normalizers at $S = 3$ and $S = 5$ end with a row normalization, while ASAP and the Sinkhorn teacher ($S = 20$) end with a column normalization.

Table H.1: Marginal errors for the same 144 frozen IMDb batch-layer cases as Table 1.

Method	Final update	$\Delta_{\mathrm{row}\to 1} \downarrow$	$\Delta_{\mathrm{col}\to 1} \downarrow$
ASAP-LS-0 ($L = 32$)	one-sided key	0.514 ± 0.053	$(1.24 \pm 0.15) \times 10^{-7}$
ASAP-KL-0 ($L = 32$)	one-sided key	0.512 ± 0.049	$(1.23 \pm 0.15) \times 10^{-7}$
ASAP-LS ($L = 32$)	two-sided column-ending	0.444 ± 0.037	$(1.27 \pm 0.15) \times 10^{-7}$
ASAP-KL ($L = 32$)	two-sided column-ending	0.444 ± 0.037	$(1.27 \pm 0.15) \times 10^{-7}$
Sinkhorn normalizer ($S = 3$)	row-ending	$(3.33 \pm 0.23) \times 10^{-7}$	0.833 ± 0.140
Sinkhorn normalizer ($S = 5$)	row-ending	$(3.13 \pm 0.13) \times 10^{-7}$	0.805 ± 0.127
Sinkhorn teacher ($S = 20$)	column-ending	0.441 ± 0.038	$(2.70 \pm 0.23) \times 10^{-7}$

H.2 ASAP-LS Slice Count Ablation

Table H.2 reports a dedicated ASAP-LS slice-count sweep on the frozen IMDb Sinkhorn teacher ($S = 20$) cases. The one-sided ASAP-LS-0 rows show the raw sliced-dual projection tradeoff, while the default two-sided ASAP-LS rows use the same fitted source duals and apply the column-ending two-sided c-transform. Increasing L improves teacher agreement for both variants and adds latency through the sliced-feature computation.

H.3 Amortization Design Ablation

Table H.3 compares design choices in the ASAP amortized layer on the frozen IMDb layers from the $S = 20$ Sinkhorn teacher study. This study varies calibration size, the one-sided and two-sided c-transform, feature family, layer sharing, and direct dual prediction.

Table H.2: ASAP-LS slice-count ablation on frozen IMDb Sinkhorn teacher ($S = 20$) layers. Entries are means over held-out batch-layer cases from three trained teachers, with $M = 4096$ fit examples. Teacher agreement is measured against the matched $S = 20$ teacher, and timing comes from the dedicated slice-count sweep pass.

Setting	Time (ms) ↓	Output RMSE ↓	Attention rel. ℓ_2 ↓	Fit time (s) ↓	Cases
ASAP-LS-0 ($L = 8,\ M = 4096$)	**5.00**	0.0278	0.483	103.6	144
ASAP-LS-0 ($L = 16,\ M = 4096$)	5.22	0.0241	0.425	103.8	144
ASAP-LS-0 ($L = 32,\ M = 4096$)	5.67	0.0201	0.364	104.2	144
ASAP-LS-0 ($L = 64,\ M = 4096$)	6.55	0.0177	0.322	104.9	144
ASAP-LS-0 ($L = 128,\ M = 4096$)	8.66	0.0158	0.289	106.6	144
ASAP-LS ($L = 8,\ M = 4096$)	8.52	0.0051	0.066	103.6	144
ASAP-LS ($L = 16,\ M = 4096$)	8.75	0.0040	0.051	103.8	144
ASAP-LS ($L = 32,\ M = 4096$)	9.20	0.0027	0.035	104.2	144
ASAP-LS ($L = 64,\ M = 4096$)	10.08	0.0018	0.024	104.9	144
ASAP-LS ($L = 128,\ M = 4096$)	12.19	**0.0014**	**0.018**	106.6	144
Sinkhorn normalizer ($S = 3$)	11.99	0.0677	1.527	–	144
Sinkhorn normalizer ($S = 5$)	17.01	0.0597	1.430	–	144
Sinkhorn teacher ($S = 20$)	54.65	–	–	–	144

Table H.3: Frozen IMDb ablations with Sinkhorn teachers ($S = 20$), reported as mean ± standard deviation over 144 held-out batch-layer cases from three teacher seeds. ASAP-0 uses the one-sided c-transform, while ASAP uses the two-sided entropic c-transform. Feature controls replace sliced potentials before the same ASAP transform, and the direct-dual row removes the analytic c-transform. Timing and speedup are measured inside this design-ablation pass, and marginal errors follow Table H.1.

Setting	Fit M	Time (ms) ↓	Output RMSE ↓	Attention rel. ℓ_2 ↓	$\Delta_{\mathrm{row}\to 1}$ ↓	$\Delta_{\mathrm{col}\to 1}$ ↓	Speedup vs. $S = 3$
ASAP-0	512	6.82 ± 0.00	0.020 ± 0.017	0.365 ± 0.255	0.514 ± 0.053	$(4.12 \pm 0.09) \times 10^{-7}$	1.76
ASAP-0	1024	6.82 ± 0.00	0.020 ± 0.017	0.365 ± 0.255	0.514 ± 0.053	$(4.12 \pm 0.09) \times 10^{-7}$	1.76
ASAP-0	4096	6.82 ± 0.00	0.020 ± 0.017	0.364 ± 0.255	0.514 ± 0.053	$(4.12 \pm 0.09) \times 10^{-7}$	1.76
ASAP	512	10.43 ± 0.00	0.003 ± 0.004	0.035 ± 0.070	0.444 ± 0.037	$(4.11 \pm 0.09) \times 10^{-7}$	1.15
ASAP	1024	10.43 ± 0.00	0.003 ± 0.004	$\mathbf{0.035 \pm 0.070}$	0.444 ± 0.037	$(4.11 \pm 0.09) \times 10^{-7}$	1.15
ASAP	4096	10.43 ± 0.00	$\mathbf{0.003 \pm 0.004}$	0.035 ± 0.070	0.444 ± 0.037	$(4.11 \pm 0.09) \times 10^{-7}$	1.15
ASAP shared coefficients	4096	10.43 ± 0.00	0.004 ± 0.006	0.051 ± 0.094	0.445 ± 0.038	$(4.11 \pm 0.09) \times 10^{-7}$	1.15
Raw projections + ASAP transform	4096	9.70 ± 0.00	0.006 ± 0.011	0.077 ± 0.167	0.447 ± 0.038	$(4.11 \pm 0.09) \times 10^{-7}$	1.24
Rank displacement + ASAP transform	4096	10.27 ± 0.00	0.006 ± 0.012	0.084 ± 0.177	0.449 ± 0.039	$(4.11 \pm 0.09) \times 10^{-7}$	1.17
Predict both duals, no c-transform	4096	7.00 ± 0.00	0.297 ± 1.191	2.767 ± 7.326	0.775 ± 0.809	1.056 ± 1.603	1.71
Sinkhorn normalizer ($S = 3$)	–	11.99 ± 0.00	0.068 ± 0.030	1.527 ± 0.319	$(3.33 \pm 0.23) \times 10^{-7}$	0.833 ± 0.140	1.00
Sinkhorn normalizer ($S = 5$)	–	17.13 ± 0.00	0.060 ± 0.015	1.430 ± 0.241	$(3.13 \pm 0.13) \times 10^{-7}$	0.805 ± 0.127	0.70
Sinkhorn teacher ($S = 20$)	–	55.59 ± 0.01	–	–	0.441 ± 0.038	$(2.70 \pm 0.23) \times 10^{-7}$	0.22

The controls support the ASAP parameterization under the same $S = 20$ teacher setting used in the main frozen-layer study. ASAP-0 is the faster one-sided operator, while the default two-sided ASAP lowers output RMSE from 0.068 for the $S = 3$ normalizer to 0.003 and remains faster than $S = 3$. Shared coefficients remain usable with a small error increase, and sliced potentials outperform raw projections and rank-displacement features under the same transform. Predicting both duals is unstable without the analytic c-transform, with attention relative ℓ_2 increasing to 2.767 and active-column error increasing to 1.056.

I IMAGENET EXPERIMENTS

We study a teacher fine-tuned from a pretrained model on ImageNet-1k and a teacher trained from random initialization on ImageNet-100. The reconstruction, resolution, and predictor studies reuse the ImageNet-1k teacher, with the evaluation subset specified for each study. All accuracies are point estimates from individual training runs.

I.1 TEACHER REPLACEMENT AND DIRECT TRAINING

For ImageNet-1k, we start from pretrained DeiT-S/16 and fine-tune for five epochs, increasing the Sinkhorn budget through $S = 3, 5, 10$. A softmax control uses the same initialization, optimizer, schedule, training budget, and data order. On all $50{,}000$ validation images, the Sinkhorn teacher reaches 77.68% and the softmax control reaches 78.38%. On the $25{,}000$-image subset used for replacement, the teacher reaches 77.512% and default ASAP-LS reaches 77.484% without final row normalization. We fit from 4096 unlabeled images and use disjoint images for fidelity evaluation. The calibration forward passes take 13–20 seconds, and the linear solve takes approximately 0.02 seconds.

For the row-normalized variant in Table 2, we normalize the reconstructed attention before multiplying by V. If $\hat{A}$ is the column-normalized reconstruction, the weights used by this variant are

$$\hat{A}^{\mathrm{row}}_{ij} = \frac{\hat{A}_{ij}}{\sum_k \hat{A}_{ik}}. \tag{I.1}$$

We evaluate the transport-plan diagnostics before this final step. The column-sum property applies to $\hat{A}$, whereas $\hat{A}^{\mathrm{row}}$ has unit row sums. The resolution sweep below omits this final normalization in the classifier as well.

On ImageNet-100, the softmax, Sinkhorn, and direct-ASAP runs use 30 training epochs from random initialization. The direct model learns ω jointly with the backbone from task labels; the calibrated model freezes the trained Sinkhorn weights and fits ω from unlabeled inputs. We evaluate all models on 5000 validation images. Direct training of the two-sided parameterization reaches 64.40%, or 63.62% with final row normalization, compared with 61.38% for the teacher and 61.28% for its calibrated replacement. The one-sided direct model reaches 51.86%. Thus the two-sided parameterization also supports joint training and gives the best accuracy in this comparison, while post-training calibration reuses an existing teacher.

I.2 ONE-SIDED AND TWO-SIDED RECONSTRUCTION

We compare ASAP-0 and ASAP using the same fitted predictor on the ImageNet-1k teacher. Table I.2 reports the relative output RMSE and KL divergence from the teacher's plan, both before final row normalization. The two-sided reconstruction reduces relative output RMSE from 0.547 to 0.0674 and KL divergence from 0.210 to 0.00688. For classification, we apply final row normalization to both reconstructions and evaluate on the same $25{,}000$ validation images. Their accuracies are closer than the attention errors suggest: 77.28% for ASAP-0 and 77.56% for ASAP, compared with 77.51% for the teacher.

Table I.1: Training comparisons on ImageNet, reported as top-1 accuracy (%) from individual runs. ImageNet-1k uses five-epoch fine-tuning of DeiT-S/16; ImageNet-100 uses 30-epoch training from scratch. Final row normalization is indicated in the row labels. Dashes denote unreported evaluations.

	ImageNet-1k		ImageNet-100
Model	Full val. (50,000)	Subset (25,000)	Val. (5000)
Softmax	78.38	–	56.82
Sinkhorn teacher ($S = 10$)	77.68	77.51	61.38
ASAP-LS, calibrated	–	77.48	–
With final row normalization	–	77.56	61.28
ASAP-0, trained from scratch	–	–	51.86
ASAP, trained from scratch	–	–	**64.40**
With final row normalization	–	–	63.62

Table I.2: One-sided and two-sided reconstruction from the same ImageNet-1k potential predictor. Attention diagnostics precede final row normalization; top-1 accuracy uses it. The teacher reaches 77.51% on the same validation subset.

Reconstruction	Relative output RMSE	Teacher-plan KL	Top-1 (%)
ASAP-0 (one-sided)	0.547	0.210	77.28
ASAP (two-sided)	0.0674	0.00688	77.56

I.3 Input Resolution

We evaluate the ImageNet-1k teacher at resolutions 224, 320, and 448, resampling its positional embeddings as the number of tokens increases. The teacher was trained at resolution 224. We refit the potential predictor using 256 calibration images at each resolution and use no final row normalization for ASAP. Table I.3 reports classification accuracy on 2000 validation images, while relative output RMSE is measured on a calibration batch. Both models lose accuracy at resolution 448, with ASAP within one percentage point of the teacher at each resolution. These measurements cover 197–785 tokens; the longer streamed-kernel runs in Table E.3 are a separate timing experiment.

Table I.3: ImageNet-1k at different input resolutions, with 256 calibration images per resolution. Accuracies use 2000 validation images; relative output RMSE uses a calibration batch. ASAP omits final row normalization.

Resolution	Tokens	Teacher (%)	ASAP (%)	Calibration output RMSE (rel.)
224	197	78.35	78.25	0.0701
320	401	77.20	77.60	0.0682
448	785	73.50	72.55	0.0650

I.4 Calibration Source

We keep the ImageNet-1k teacher fixed and change the images used to fit its potentials. The calibration images come from either ImageNet or COCO (Lin et al., 2014); evaluation uses 64 images from the target dataset. In each direction, we compare cross-dataset calibration with calibration from the target dataset at the same M. Table I.4 reports the relative increase in output error for default ASAP. The increases range from 2.1% to 4.6% across the measured sizes. COCO is used here only as an image source for calibration and attention evaluation, not as a detection or segmentation benchmark.

Table I.4: Increase in relative output error (%) after changing the calibration dataset. Each entry compares with in-domain calibration on the same evaluation dataset and the same frozen ImageNet teacher.

Calibration → evaluation	$M = 128$	$M = 512$	$M = 2048$
ImageNet → COCO	$+3.3$	$+2.5$	$+2.1$
COCO → ImageNet	$+4.6$	$+2.8$	$+2.9$

I.5 Predictor Alternatives

The predictor study uses the ImageNet-1k teacher, 2048 calibration images, and 128 disjoint images for fidelity evaluation. The sliced linear baseline fits 32 coefficients per head, for 2304 coefficients across the model. This is a head-specific fit, rather than the head-shared fit used in the compact IMDb configuration. We compare richer feature dictionaries and an MLP with the same attention reconstruction. Table I.5 reports error for the one-sided and two-sided reconstructions, predictor cost at $N = 197$, and the additional parameter counts.

Table I.5: Predictor alternatives on the ImageNet-1k teacher. Error is relative output RMSE before final row normalization. Cost is measured in units of one column-normalization pass at $N = 197$. RFF denotes random Fourier features; its row uses the best of three tested bandwidths.

Predictor	Features	ASAP-0 error	ASAP error	Cost	Parameters
Sliced linear	32	0.550	0.067	5.59	2304
Sliced linear + two features	34	0.224	0.037	6.96	2448
RFF	128	0.540	0.059	5.95	59904
Nyström-RBF on sliced features	255	0.513	0.055	12.1	$\sim$ 900k
Nyström-RBF on queries	256	0.208	0.036	7.57	$\sim$ 1.0M
MLP ($32 \to 64 \to 1$)	32	0.500	0.054	6.47	156744
Sliced linear, more slices	256	0.305	0.035	39.6	18432

The two-feature extension adds the row log-sum-exp of the score matrix and allows the predictor to learn the coefficient of the query-norm shift used in the score-to-cost conversion. With the two-sided reconstruction, this extension reduces error from 0.067 to 0.037 with 144 additional coefficients. At $N = 197$, its predictor cost rises from 5.59 to 6.96 in units of one column-normalization pass. The RBF and MLP predictors also reduce error, and increasing the number of slices to 256 reaches 0.035 at a higher measured cost. The cost comparison depends on length: a column-normalization pass is quadratic in N, whereas the feature evaluation has a different scaling. We therefore use the $N = 197$ costs to compare these predictors at that length, not as fixed costs relative to normalization at all lengths.