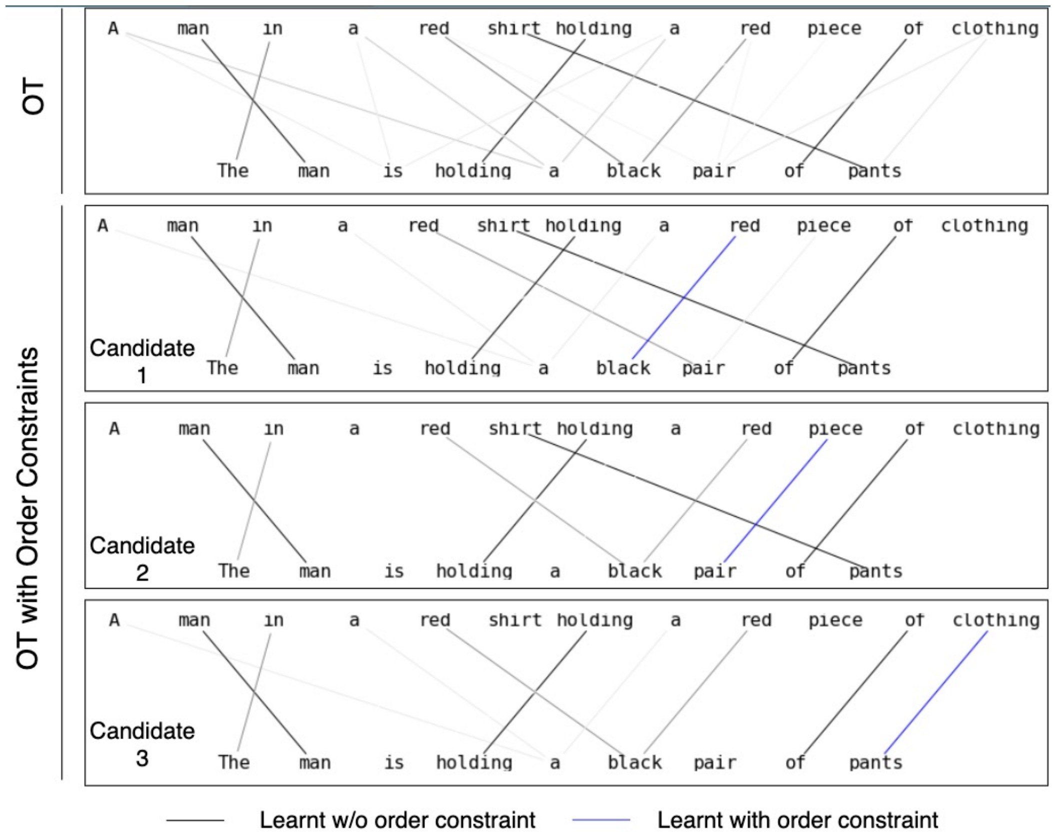
Order Constraints in Optimal Transport

Fabian Lim, Laura Wynter, Shiau Hong Lim

IBM Research, Singapore

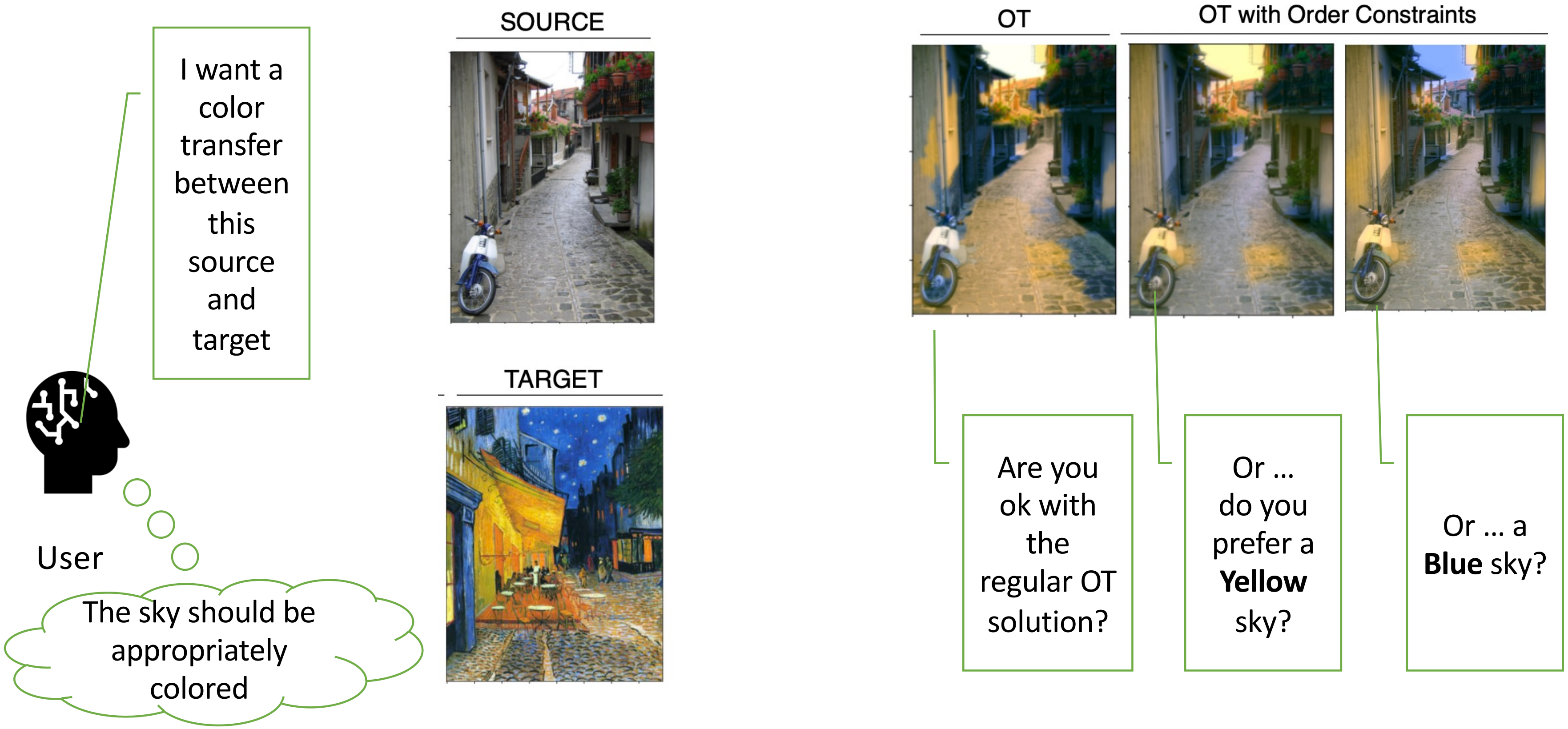
Optimal Transport with Order Constraints

- Provide **context** to OT
- Example context in text application
  - “red”-”black” (color)
  - ”piece”-”pair” (multiplicity)
  - “clothing”-”pants” (inventory)
- Standard OT gives top solution; OT with OC gives all 4.
- The human user selects the most interpretable solution.



Optimal Transport with Order Constraints

- Consider color transfer



Formulation of OT with Order Constraints

- For given sum/row weights *a,b*, OT minimizes a linear cost *D* over a transport polytope:

$$U(a,b) = \{\Pi \in \mathbb{R}_+^{m\times n} : \Pi \mathbf{1}_n = a, \Pi^T \mathbf{1}_m = b\}$$

- **We propose to define** **order constraints (OC)** **on top of the transport polytope**:

$$\min_{\Pi \in U(a,b)} f(\Pi) := \operatorname{tr}\left(D^T \Pi\right),$$

$$\text{s.t } \Pi_{i_k j_k} \geq \cdots \geq \Pi_{i_1 j_1}, \quad \Pi_{i_1 j_1} \geq \Pi_{pq} \text{ for } pq \in V,$$

mn inequalities.

where *V* indexes outside of *k* num. of OC positions:

$$V := [mn] \setminus \{i_\ell j_\ell : \ell \in [k]\}$$

Prop. 2.1, Cor 2.2 provides sufficient conditions for the OT with OC to be feasible. In general, feasibility depends on constraints *a,b and OC positions.*

Solving the OT with OC Formulation

- Propose an iterative ADMM formulation

Algorithm 1 Iterative procedure for OT under order constraints $O_{ij_{[k]}}$ with linear costs $f(X) = \mathrm{tr}\left(D^T X\right)$.

Require: Costs D, penalty ρ, initial X_0, Z_0.

1: **for** round $t \geq 1$ until stopping **do**
2: Update $X_{t+1} = \mathrm{Proj}_{\mathcal{C}_1(a,b)}\left(Z_t - M_t - \rho^{-1} D\right)$
3: Update $Z_{t+1} = \mathrm{Proj}_{\mathcal{C}_2}\left(X_{t+1} + M_t\right)$.
4: Update (scaled) dual variable $M_{t+1} = M_t + X_t - Z_t$.
5: Return X_t

- Line 2 $\longrightarrow$ $\mathcal{C}_1(a, b)$ Proposition 3.1 (Lim, et. al)
  Affine, matrix-vector ops

Solving the OT with OC Formulation

- Alg1, Line 3: Update $Z_{t+1} = \text{Proj}_{\mathcal{C}_2}(X_{t+1} + M_t)$ $\longrightarrow$ $\mathcal{C}_2$

Proposition 3.2 (Lim, et. al)
Extension *of Pooled-Adjacent Violators Algorithm (PAVA), (Grotzinger, Witzgall, 84),*

Algorithm 2 ePAVA for $\mathcal{C}_2 = O_{ij_{[k]}}$ for $k \in [mn]$

Require: $X \in \mathbb{R}^{m \times n}$. Indices $ij_{[k]}$.

$\ell := 1, B := 1,$ `le`$[1] :=$ `ri`$[1] := 1,$ `val`$[1] := T(0)$.
for $\ell \leq k$ **do**
  $B := B + 1, \ell := \ell + 1,$ `le`$[B] :=$ `ri`$[B] := \ell,$ `val`$[B] := X_{i_\ell j_\ell}$.
  for $B \geq 2$ and `val`$[B] \leq$ `val`$[B-1]$ **do**
    Let $q =$ `ri`$[B]$.
    if $B = 2$ **then**
      Solve and store $\tilde{\eta} \geq 0$ satisfying $T(\tilde{\eta}) = \Delta_{2q} + \tilde{\eta}/(q-1)$. Set `val`$[B-1] := T(\tilde{\eta})$.
    else
      Set `val`$[B-1] := \Delta_{pq}$ for $p =$ `ri`$[B-1]$.
    Set `ri`$[B-1] :=$ `ri`$[B]$. Decrement $B := B - 1$.
Return $B, \tilde{\eta},$ `le`, `ri`, and `val`.

coalescing

Threshold T

$$T(\cdot) := (\tau(t(\eta), \eta))_+$$

$$\tau(s, \eta) = \frac{1}{s+1}\left(X_{i_1 j_1} - \eta + \sum_{\ell=1}^{s} X_{(\ell)}\right),$$

$$t(\eta) + 1 = \arg\min\left\{s \in [r] : \tau(s, \eta) > X_{(s)}\right\}$$

Threshold Δ

$$\Delta_{pq} := \sum_{\ell=p}^{q} X_{i_\ell j_\ell} / (q - p + 1)$$

Solving the OT with OC Formulation

- For a problem size *mn*, Algorithm 1, initialized with $Z_0 = M_0 = 0$, achieves iteration error:

$$f(\bar{X}_t) - f^* \leq \delta,$$

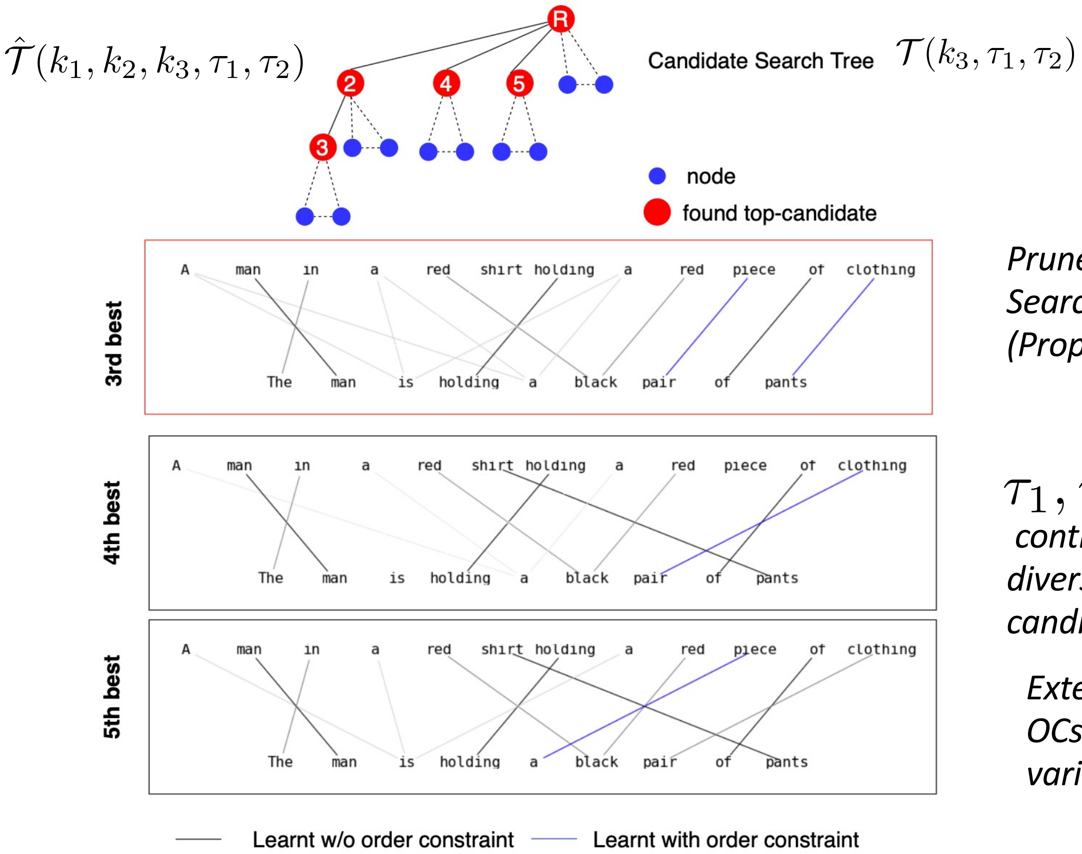
in at most the following number of operations:

$$\mathcal{O}(\|D\|_\infty/\delta \cdot mn\log(mn))$$

Thm 3.3, Prop 3.4 (Lim, et. al)

Explainability via Branch-and-Bound

- Estimate important variates to define order constraints



Prune Search (Prop. 4.1)

τ_1, τ_2 control diversity of candidates

Extend k OCs by variate ij

Algorithm 3 Learning subtree $\widehat{\mathcal{T}}(k_1, k_2, k_3, \tau_1, \tau_2)$ of $\mathcal{T}(k_3, \tau_1, \tau_2)$ and top-k_2 candidate plans for linear costs $f(\Pi) = \text{tr}\left(D^T \Pi\right)$.

Require: Costs D, thresholds $0 \leq \tau_1, \tau_2 \leq 1$. Search upper limit k_1, number of top candidates k_2, and search depth $k_3 \leq \min(m, n)$.

1: Compute $\hat{\Pi}_1$ using (1). Init $\widehat{\mathcal{T}}(k_1, k_2, k_3, \tau_1, \tau_2)$.
2: Use $\hat{\Pi}_1, \tau_1, \tau_2$ in (16) and (17) to obtain $\mathcal{I}$ and Φ_{ij}. Init. $\mathcal{S} = \{(ij, \Phi_{ij}) : ij \in \mathcal{I}\}$. `count=0`.
3: **for** `count` $< k_1$ **do**
4: Pop $ij_{[k]}$ having smallest Φ in $\mathcal{S}$, for some k constraints. Compute $\mathcal{L}$ from right-hand side of (23).
5: **if** $\hat{\Pi}_{k_2}$ is not yet obtained or $\mathcal{L} > f(\hat{\Pi}_{k_2})$ **then**
6: Solve Algorithm 1 with order constraint $O_{ij_{[k]}}$ for new candidate $\hat{\Pi}$. Set `count += 1`.
7: Update top-k_2 candidates $\hat{\Pi}_1, \hat{\Pi}_2, \cdots, \hat{\Pi}_{k_2}$ and $\widehat{\mathcal{T}}(k_1, k_2, k_3, \tau_1, \tau_2)$ using new candidate $\hat{\Pi}$.
8: **if** k equals k_3 **then**
9: Go to line 4.
10: **if** $\hat{\Pi}_{k_2}$ not yet obtained or $f(\hat{\Pi}) < f(\hat{\Pi}_{k_2})$ **then**
11: Use $\hat{\Pi}_1, \tau_1, \tau_2$ in (16) and (17) and obtain new variates $ij \in \mathcal{I}(\hat{\Pi})$ and $\{\Phi_{ij}\}_{ij \in \mathcal{I}(\hat{\Pi})}$.
12: **for** variate ij in $\mathcal{I}(\hat{\Pi})$ **do**
13: **if** $i \notin i_{[k]}$ and $j \notin j_{[k]}$ **then**
14: Push $(ij, i_1 j_1, \cdots, i_k j_k, \Phi_{ij})$ onto stack $\mathcal{S}$.
15: Return top k_2 candidates $\hat{\Pi}_1, \hat{\Pi}_2, \cdots, \hat{\Pi}_{k_2}$ and $\widehat{\mathcal{T}}(k_1, k_2, k_3, \tau_1, \tau_2)$.

Experiment Results

BESTF1 is a measure of how well candidate solutions agree with human-annotations in e-SNLI dataset

- Sentence Relationship Classification (e-SNLI)
  - OT with OC provides up to 5 points improvement over Greedy search

Algorithm	BestF1@n		
	n= 2	n= 5	n= 10
Algorithm 3 (ours)	68.1 ± .2	71.2 ± .3	73.7 ± .2
Greedy version	67.9 ± .3	68.2 ± .3	68.2 ± .3

- Color Transfer (SUN, WikiArt)
  - OT results have problems
  - Prior (human-crafted) constraints try to correct observed problems
  - OT w/ OC produces solutions that perform similarly as prior(human-crafted) versions