A PROOFS

Let $\mathbb{Y}$ be the output space, $\mathrm{y}_i, \mathrm{y}_j, \mathrm{y}_k \in \mathbb{Y}$, and $\boldsymbol{y}_k \in \mathbb{Y} - \mathrm{y}_i - \mathrm{y}_j$ be a subset of the symbols excluding $\mathrm{y}_i, \mathrm{y}_j$.

Lemma A.1 $\mathrm{y}_i \not\perp\!\!\!\perp \mathrm{y}_j \implies \mathrm{y}_i \not\perp\!\!\!\perp (\mathrm{y}_j \mathrm{y}_k)$

Proof Let $\mathrm{y}_i \perp\!\!\!\perp (\mathrm{y}_j \mathrm{y}_k)$ by contradiction. Then:

$$p(\mathrm{y}_i, \mathrm{y}_j \mathrm{y}_k) = p(\mathrm{y}_i) p(\mathrm{y}_j \mathrm{y}_k) \tag{2}$$

Also,

$$\begin{aligned} p(\mathrm{y}_i, \mathrm{y}_j) &= \sum_{\mathrm{y}_k \in \boldsymbol{Z}} p(\mathrm{y}_i, \mathrm{y}_j \mathrm{y}_k) \\ &= \sum_{\mathrm{y}_k \in \boldsymbol{Z}} p(\mathrm{y}_i) p(\mathrm{y}_j \mathrm{y}_k) && \text{(equation 2)} \\ &= p(\mathrm{y}_i) \sum_{\mathrm{y}_k \in \boldsymbol{Z}} p(\mathrm{y}_j \mathrm{y}_k) \\ &= p(\mathrm{y}_i) p(\mathrm{y}_j) && (3) \end{aligned}$$

However, $\mathrm{y}_i \not\perp\!\!\!\perp \mathrm{y}$ thus $\mathrm{y}_i \not\perp\!\!\!\perp \mathrm{y} \implies \mathrm{y}_i \not\perp\!\!\!\perp (\mathrm{y}_j \mathrm{y}_k)$.

Lemma A.2

$$p(\mathrm{y}_i \mid \mathrm{y}_j) > p(\mathrm{y}_j \mid \mathrm{y}_i) \implies p(\mathrm{y}_i \mid \mathrm{y}_j, \boldsymbol{y}_k) > p(\mathrm{y}_j \mid \mathrm{y}_i, \boldsymbol{y}_k)$$

if $\mathrm{y}_i, \mathrm{y}_j \perp\!\!\!\perp \boldsymbol{y}_k$

Proof We have:

$$\begin{aligned} & p(\mathrm{y}_i \mid \mathrm{y}_j) > p(\mathrm{y}_j \mid \mathrm{y}_i) \\ \implies & p(\mathrm{y}_j) < p(\mathrm{y}_i) && (4) \end{aligned}$$

$$\begin{aligned} p(\mathrm{y}_j, \boldsymbol{y}_k) &= p(\boldsymbol{y}_k \mid \mathrm{y}_j) p(\mathrm{y}_j) \\ &< p(\boldsymbol{y}_k \mid \mathrm{y}_j) p(\mathrm{y}_i) && \text{(Equation 4)} \\ &= p(\boldsymbol{y}_k \mid \mathrm{y}_i) p(\mathrm{y}_i) && (\mathrm{y}_i, \mathrm{y}_j \perp\!\!\!\perp \boldsymbol{y}_k \implies p(\boldsymbol{y}_k \mid \mathrm{y}_j) = p(\boldsymbol{y}_k \mid \mathrm{y}_i) = p(\boldsymbol{y}_k)) \\ &= p(\mathrm{y}_i, \boldsymbol{y}_k) && (5) \end{aligned}$$

Thus,

$$\begin{aligned} p(\mathrm{y}_i \mid \mathrm{y}_j, \boldsymbol{y}_k) &= \frac{p(\mathrm{y}_i, \mathrm{y}_j, \boldsymbol{y}_k)}{p(\mathrm{y}_j, \boldsymbol{y}_k)} \\ &> \frac{p(\mathrm{y}_i, \mathrm{y}_j, \boldsymbol{y}_k)}{p(\mathrm{y}_i, \boldsymbol{y}_k)} \\ &= p(\mathrm{y}_j \mid \mathrm{y}_i, \boldsymbol{y}_k) && (6) \end{aligned}$$

Lemma A.3 *If* $\mathrm{y}_i \perp\!\!\!\perp \mathrm{y}_j \; \forall \mathrm{y}_i, \mathrm{y}_j \in \mathbb{Y}$*, the order is guaranteed to not affect learning.*

Proof Let π_j be the j^{th} order over $\mathbb{Y}$ (out of $|\mathbb{Y}|!$ possible orders Π), and $\pi_j(\mathbb{Y})$ be the sequence of elements in $\mathbb{Y}$ arranged with π_j.

$$\begin{aligned} & p(\mathrm{y}_i \mid \mathrm{y}_j) = p(\mathrm{y}_i) && (\mathrm{y}_i \perp\!\!\!\perp \mathrm{y}_j \; \forall \mathrm{y}_i, \mathrm{y}_j) \\ \implies & p(\mathrm{y}_i, \mathrm{y}_j, \mathrm{y}_k) = p(\mathrm{y}_i) p(\mathrm{y}_j \mid \mathrm{y}_i) p(\mathrm{y}_k \mid \mathrm{y}_i, \mathrm{y}_j) \\ & = p(\mathrm{y}_i) p(\mathrm{y}_j) p(\mathrm{y}_k) \\ \implies & p(\pi_m(\mathrm{y}_i, \mathrm{y}_j, \mathrm{y}_k)) = p(\pi_n(\mathrm{y}_i, \mathrm{y}_j, \mathrm{y}_k)) \; \forall \pi_m, \pi_m \in \Pi \end{aligned}$$

In other words, when all elements are mutually independent, all possible joint factorizations will simply be a product of the marginals, and thus identical.

B DATASET

	Input	Output
Fine-grained emotion classification, [28] (Demszky et al., 2020)	*So there's hope for the rest of us! Thanks for sharing. What helped you get to where you are?*	{curiosity, gratitude, optimism}
Open-entity typing [2519] (Choi et al., 2018)	*Some 700,000 cubic meters of caustic sludge and water burst inundating [SPAN] three west Hungarian villages [SPAN] and spilling.*	{colony, region, location, hamlet, area, village, settlement, community}
Reuters [90] (Lewis, 1997)	*India is reported to have bought two white sugar cargoes for... ...cargo sale, they said.*	{ship, sugar}

Table 3: Real world tasks used for experiments

C FIXING THE PROPOSAL DISTRIBUTION IN THE VAE FORMULATION

$$
\begin{aligned}
\log p_\theta(\mathbb{Y} \mid \boldsymbol{x}) &= \log \sum_{\pi_{\boldsymbol{z}} \in \Pi} p_\theta(\pi_z(\mathbb{Y}) \mid \boldsymbol{x}) \\
&= \log \sum_{\pi_{\boldsymbol{z}} \in \Pi} \frac{q_\phi(\pi_{\boldsymbol{z}})}{q_\phi(\pi_{\boldsymbol{z}})} p_\theta(\pi_z(\mathbb{Y}) \mid \boldsymbol{x}) \\
&= \log \mathbb{E}_{q_\phi(\pi_{\boldsymbol{z}})} \left[\frac{p_\theta(\pi_z(\mathbb{Y}) \mid \boldsymbol{x})}{q_\phi(\pi_{\boldsymbol{z}})} \right] \\
&\geq \mathbb{E}_{q_\phi(\pi_{\boldsymbol{z}})} \left[\log p_\theta(\mathbb{Y}, \pi_{\boldsymbol{z}} \mid \boldsymbol{x}) \right] - \mathbb{E}_{q_\phi(\pi_{\boldsymbol{z}})} \left[\log q_\phi(\pi_{\boldsymbol{z}}) \right] \\
\log p_\theta(\mathbb{Y} \mid \boldsymbol{x}) = \log \sum_{\pi_{\boldsymbol{z}} \in \Pi} p_\theta(\pi_z(\mathbb{Y}) \mid \boldsymbol{x}) &\geq \underbrace{\mathbb{E}_{q_\phi(\pi_{\boldsymbol{z}})} \left[\frac{\log p_\theta(\pi_{\boldsymbol{z}}(\mathbb{Y}) \mid \boldsymbol{x})}{q_\phi(\pi_{\boldsymbol{z}})} \right]}_{\text{ELBO}} = \mathcal{L}(\theta, \phi)
\end{aligned}
\tag{7}
$$

Where equation 7 is the evidence lower bound (ELBO). The success of this formulation depends on the quality of the proposal distribution q from which the orders are drawn. When q is fixed (e.g., to uniform distribution over the orders), learning only happens for θ. This can be clearly seen from splitting Equation 7 into terms that involve just θ and ϕ:

$$
\begin{aligned}
\nabla_\phi \mathcal{L}(\theta, \phi) &= 0 \\
\nabla_\theta \mathcal{L}(\theta, \phi) &= \nabla_\theta \mathbb{E}_{q_\phi(\pi_{\boldsymbol{z}})} \left[\log p_\theta(\mathbb{Y}, \pi_{\boldsymbol{z}} \mid \boldsymbol{x}) \right]
\end{aligned}
$$

D HYPERPARAMETERS

We list all the hyperparameters in Table 4.

E EXPLORING THE INFLUENCE OF ORDER ON SEQ2SEQ MODELS WITH A SIMULATION

We design a simulation to investigate the effects of output order and cardinality on conditional set generation, following prior work that has found simulation to be an effective for studying properties of deep neural networks (Vinyals et al., 2016; Khandelwal et al., 2018).

Hyperparameter	Value
GPU	GeForce RTX 2080 Ti
gpus	1
auto_select_gpus	false
accumulate_grad_batches	1
max_epochs	3
precision	32
learning_rate	1e-05
adam_epsilon	1e-08
num_workers	16
warmup_prop	0.1
seeds	[15143, 27122, 999888]
add_lr_scheduler	true
lr_scheduler	linear
max_source_length	120
max_target_length	120
val_max_target_length	120
test_max_target_length	120

Table 4: List of hyperparameters used for all the experiments.

Data generation We use a graphical model (Figure 3) to generate conditionally dependent pairs $(\boldsymbol{x}, \mathbb{Y})$, with different levels of interdependencies among the labels in $\mathbb{Y}$. Let $\mathbb{Y} = \{\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_n\}$ be the set of output labels. We sample a dataset of the form $\{(\boldsymbol{x}, \boldsymbol{y})\}_{i=1}^{m}$. $\boldsymbol{x}$ is an n dimensional multinomial sampled from a dirichlet parameterized by α, and $\boldsymbol{y}$ is a sequence of symbols with each $y_i \in \mathbb{Y}$. The output sequence $\boldsymbol{y}$ is created in B *blocks*, each block of size k. A block is created by first sampling $k-1$ prefix symbols independently from Multinomial($\boldsymbol{x}$), denoted by $\boldsymbol{y}_p$ The k^{th} suffix symbol (y_s) is sampled from either a uniform distribution with a probability = ϵ or is deterministically determined from the preceding $k-1$ prefix terms. For block size of 1 ($k=1$), the output is simply a set of size B sampled from $\boldsymbol{x}$ (i.e., all the elements are independent). Similarly, $k=2$ simulates a situation with a high degree of dependence: each block is of size 2, with the prefix sampled independently from the input, and the suffix determined deterministically from the prefix. Gradually increasing the block size increases the number of independent elements.

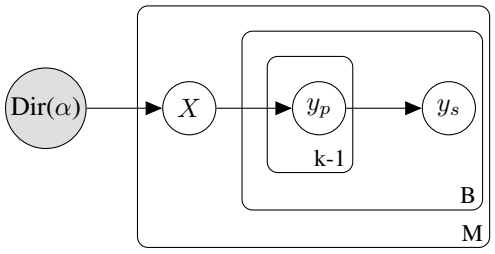

Figure 6: The generative process for simulation

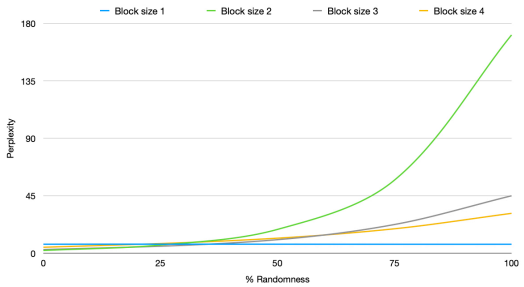

Figure 7: Perplexity vs. Randomness for varying block sizes

E.1 Major Findings

We now outline our findings from the simulation. We use the architecture of BART-base Lewis et al. (2020) (six-layers of encoder and decoder) without pre-training for all simulations. All the simulations were repeated using three different random seeds, and we report the averages.

Finding 1: SEQ2SEQ models are sensitive to order, but only if the labels are conditionally dependent on each other. We train with the prefix $\boldsymbol{y}_p$ listed in the lexicographic order. At test time, the order of is randomized from 0% (same order as training) to 100 (appendixly shuffled).

As can be seen from Figure 7 the perplexity gradually increases with the degree of randomness. Further, note that perplexity is an artifact of the model and is independent of the sampling strategy used, showing that order affects learning.

Finding 2: Training with random orders makes the model less sensitive to order As Figure 8 shows, augmenting with random order makes the model less sensitive to order. Further, augmenting with random order keeps helping as the perplexity gradually falls, and the drop shows no signs of flattening.

Finding 3: Effects of position embeddings can be overcome by augmenting with a sufficient number of random samples Figure 8 shows that while disabling position embedding helps the baseline, similar effects are soon achieved by increasing the random order. This shows that disabling position embeddings can indeed alleviate some concerns about the order. This is crucial for pre-trained models, for which position embeddings cannot be ignored.

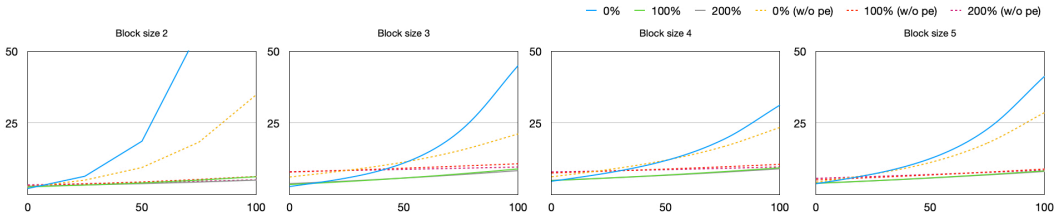

Figure 8: Augmenting dataset with multiple orders help across block sizes. Augmentations also overcome any benefit that is obtained by using position embeddings.

Finding 4: TSAMPLE leads to higher set overlap We next consider blocks of order 2 where the prefix symbol y_p is selected randomly as before, but the suffix is set to a special character y'_p with 50% probability. As the special symbol y'_p only occurs with y_p, there is a high pmi between each (y_p, y'_p) pair as $p(y_p \mid y'_p) = 1$. Different from finding 1, the output symbols are now shuffled to mimic a realistic setup. We gradually augment the training data with random and topological orders and evaluate the learning and the final set overlap using training perplexity and Jaccard score, respectively. The results are shown in Figure 9. Similar trends hold for larger block sizes, and the results are included in the Appendix in the interest of space.

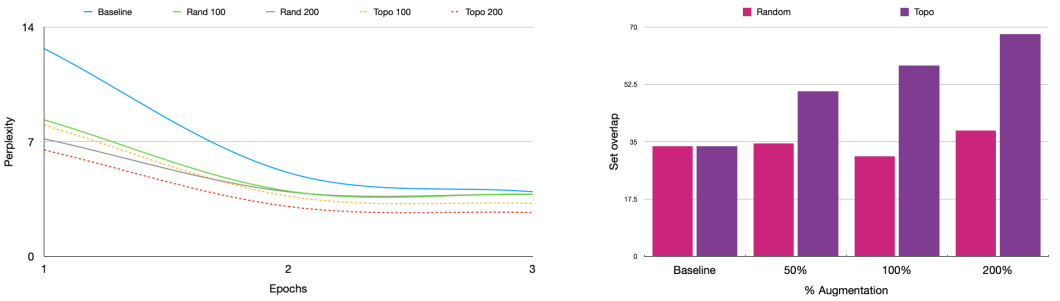

Figure 9: Effect of TSAMPLE on perplexity and set overlap. **Left:** Augmentations done TSAMPLE helps the model converge faster and to a lower perplexity. **Right:** Using TSAMPLE, the overlap between training and test set increases consistently, while consistently outperforming UNIFORM.

Finding 5: TSAMPLE helps across all sampling types We see from Table 5 that our approach is not sensitive to the sampling type used. Across five different sampling types, augmenting with topological orders yields significant gains.

Finding 6: SEQ2SEQ models can learn cardinality and use it for better decoding We created sample data from Figure 6 where the length of the output is determined by sum of the inputs X.

	Beam	Random	Greedy	Top-k	Nucleus
UNIFORM	0.39 ± 0.05	0.39 ± 0.02	0.35 ± 0.05	0.39 ± 0.02	0.39 ± 0.02
TSAMPLE	0.67 ± 0.05	0.67 ± 0.05	0.71 ± 0.04	0.67 ± 0.05	0.68 ± 0.05

Table 5: Set overlap for different sampling types with 200% augmentations. The gains are consistent across sampling types. Similar trends were observed for 100% augmentation and without positional embeddings. Top-k sampling was introduced by (Fan et al., 2018), and Nucleus sampling by (Holtzman et al., 2020).

We experimented with and without including cardinality as the first element. We found that training with cardinality increases step overlap by over 13%, from 40.54 to 46.13. Further, the version with cardinality accurately generated sets which had the same length as the target 70.64% of the times, as opposed to 27.45% for the version without cardinality.

F ADDITIONAL RESULTS

We present all the results for the three tasks in Tables 6, 7, and 8.

	p_{micro}	p_{macro}	r_{micro}	r_{macro}	F_{micro}	F_{macro}	$jaccard$
SET SEARCH	47.17	10.68	13.09	7.02	10.7	7.36	7.4
SEQ2SEQ	41.65	27.39	35.19	26.21	27.4	23.41	23.4
SEQ2SEQ + CARD	39.77	33	38.02	28.31	33	26.79	26.8
UNIFORM + CARD	44.77	35.6	32.96	26.54	35.6	27.53	27.5
TSAMPLE + CARD	43.37	36.08	34.51	30.54	36.1	30.01	30
UNIFORM- CARD	48.85	32.45	27.75	19.86	32.5	22.67	22.7
TSAMPLE- CARD	50	36.68	29.84	19.84	36.7	23.31	23.3

Table 6: Results for GO-EMO.

	p_{micro}	p_{macro}	r_{micro}	r_{macro}	F_{micro}	F_{macro}	$jaccard$
SET SEARCH	70.04	10.92	34.9	7.1	46.56	7.54	37.49
SEQ2SEQ	66.36	24.74	42.28	13.78	51.64	15.58	44.3
SEQ2SEQ + CARD	73.02	34.17	53.8	21.85	61.95	24.28	59.08
UNIFORM + CARD	74.26	35.31	54.33	22.13	62.75	24.74	58.95
TSAMPLE + CARD	75.65	36.67	55.54	24.13	64.05	26.66	61.14
UNIFORM- CARD	69.56	26.68	38.15	12.71	49.27	15.2	42.24
TSAMPLE- CARD	76.55	26.49	41.78	12.77	54.06	15.78	47.34

Table 7: Results for REUTERS.

	p_{micro}	p_{macro}	r_{micro}	r_{macro}	F_{micro}	F_{macro}	$jaccard$
SET SEARCH	24.65	26.5	29.98	31.44	23.92	26.25	13.39
SEQ2SEQ	52.78	55.4	39.84	42.42	41.45	44.63	24.6
SEQ2SEQ + CARD	61.26	62.48	41.87	44.68	48.07	50.48	27.84
UNIFORM + CARD	67.56	68.59	39.61	42.25	47.98	50.4	26.89
TSAMPLE + CARD	64.58	65.53	44.6	47.46	51.2	53.48	29.39
UNIFORM- CARD	60.93	62.57	39.09	41.69	44.2	46.85	25.26
TSAMPLE- CARD	58.02	59.88	42.63	44.95	46.54	48.86	26.82

Table 8: Results for OPENENT.