Structure-Conditional Minimum Bayes Risk Decoding

Anonymous ACL submission

Abstract

Minimum Bayes Risk (MBR) decoding has seen renewed interest as an alternative to traditional generation strategies. While MBR has proven effective in machine translation, where the variability of a language model's outcome space is naturally constrained, it may face challenges in more open-ended tasks such as dialogue or instruction-following. We hypothesise that in such settings, applying MBR with standard similarity-based utility functions may result in selecting responses that are broadly representative of the model's distribution, yet suboptimal with respect to any particular grouping of generations that share an underlying *latent structure*. In this work, we introduce three lightweight adaptations to the utility, designed to make MBR more sensitive to structural variability in the outcome space. To test our hypothesis, we curate a dataset capturing three representative types of latent structure: dialogue act, emotion, and response structure. We also propose two metrics to evaluate the structural optimality of MBR. Our analysis demonstrates that common utility functions fall short by these metrics. In contrast, our proposed adaptations considerably improve structural optimality. Finally, we evaluate our approaches on real-world instruction-following benchmarks, AlpacaEval and MT-Bench, and show that increased structural sensitivity improves generation quality by up to 13.7 percentage points in win rate.

1 Introduction

Once a language model has been trained, one fundamental problem remains: determining how to select an output sequence from the model's learned probability distribution over possible continuations. Traditional approaches such as beam search decoding and majority voting aim to select a high probability continuation under the model distribution. However, a growing body of research has shown model probability to not reliably align with human preferences (Stahlberg and Byrne, 2019; Zhang et al., 2021) and, in response, Minimum Bayes Risk (MBR; Kumar and Byrne, 2004; Eikema and Aziz, 2020) decoding has emerged as a more robust alternative. MBR casts decoding as a decision-theoretic problem, selecting the output that minimises risk with respect to a task-specific utility function, under the uncertainty over continuations represented by the language model. This utility typically reflects the degree of agreement between a candidate and the broader set of outcomes, penalising those that diverge significantly from the consensus. By integrating both model probabilities and inter-candidate consistency, MBR yields generations that are better aligned with human preferences, regularly outperforming conventional methods (Freitag et al., 2022; Wu et al., 2025).

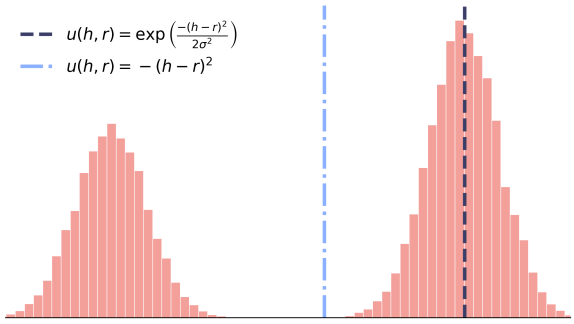

Figure 1: The choice of utility function can considerably impact the Minimum Bayes Risk (MBR) optimum. When the outcome space is structured or multimodal, the MBR optimum may settle between modes, landing in a region of low probability. Here, we present a continuous example featuring a bimodal Gaussian distribution and show the MBR optima (vertical lines) of two utility functions with markedly different behaviours.

MBR decoding has gained significant attention in neural machine translation, where utility is often measured by task-agnostic sentence similarity scores. This corresponds to selecting the sequence which, in expectation and under the lens of a particular similarity score, most closely matches the broader distribution of sequences prescribed by the

model. While this decoding strategy works well in translation—where outcome space variability is inherently constrained by the task—it risks being less effective for tasks with a broader range of contextually plausible *latent structures*, and thus greater variability in realisations, such as dialogue or instruction-following (Giulianelli et al., 2023). Consider, for instance, the following dialogue exchange: **A:** *The mountains would be a great place for the lab retreat.* **B:** *That's a wonderful choice.* In response, speaker A could follow up with a statement (*The mountains offer many outdoor team-building activities.*), a question (*Which aspects of the mountains are you most excited about?*), an instruction (*Please check out different venues online to finalise the decision.*), or an offer (*Shall I make the necessary arrangements?*). Similarly, when given an instruction like *Please summarise Gödel, Escher, Bach*, valid responses could range from a single-sentence summary, to a paragraph with more detail, a multi-paragraph narrative, or even a list of key topics. In such settings, applying MBR with a standard similarity-based utility function may result in selecting an output that is broadly representative of the model's outcome distribution, but suboptimal with respect to any one plausible latent structure (we illustrate this in Fig. 1 using a continuous distribution as a simplified example).

In this work, we propose adapting utility functions for MBR such that they are able to explicitly account for a language model's uncertainty over latent structures. We adopt a broad interpretation of *structure*, treating it as a latent variable that influences the form a generation takes—for example, a dialogue act, the level of detail in a response, or the emotion conveyed by an utterance. To examine how reliably MBR selects the highest-consistency candidate within clusters of generations that share a latent structure—what we call *cluster-optimality*, we semi-automatically construct a dataset of 3,000 curated outcome spaces, for a total of 350,000 candidate generations. These are conditioned on naturally occurring conversational and instruction-following contexts, but present controlled uncertainty over three types of structure: dialogue act, emotion, and response structure (*i.e.*, a single sentence, a paragraph, a list, or a table). Our analysis of this dataset shows that, under commonly used utility functions, MBR solutions are cluster-optimal *in fewer than half of the cases*. To address this, we introduce three new approaches—Clustering, Structure Embeddings, and Utility Cutoff—that adapt utility functions to account for a candidate's (soft) membership in structure-specific candidate groups, while preserving the decision-theoretic foundation of risk minimisation.

Our experiments confirm that adapting the utility function to account for latent structural variability substantially improves MBR solutions. On our curated dataset with controlled uncertainty over dialogue act, emotion, and response structure, our three proposed methods achieve markedly higher cluster optimality than standard MBR with BERTScore or BLEURT utilities. We also observe gains on real-world instruction-following benchmarks, despite the absence of explicit structure annotations. In particular, our methods improve generation quality on AlpacaEval and MT-Bench, with win rates against GPT-4o increasing by up to 13.7 percentage points on the latter. These findings support our central claim: structure-aware utility functions enable MBR to more reliably select high-quality outputs in tasks where structural variability is inherent to the generation space.

2 Language Modeling and Decision Rules

A language model P is a distribution over strings Σ^*, where Σ is an alphabet, *i.e.*, a finite, non-empty set of symbols, and Σ^* its Kleene closure, *i.e.*, the set of all strings formed by concatenating symbols in Σ, including the empty string ε. We define Y as a random variable over sequences within Σ^*. Every language model can be expressed in autoregressive form by decomposing the probability of a string as the product of conditional probabilities of each symbol, followed by an end-of-string event EOS:

$$P(Y = y) = P(\text{EOS} \mid y) \prod_{t=1}^{|y|} P(y_t | y_{<t}), \quad (1)$$

where each conditional distribution $P(Y_t \mid y_{<t})$ is a probability distribution over $\Sigma \cup \{\text{EOS}\}$. This formulation underlies most modern autoregressive language models, where each conditional probability is produced by a learned parametric model. We will assume an implicit conditioning on some set of neural network parameters θ estimated during neural network training on some dataset. Furthermore, a language model is commonly conditioned on an input, or a prompt, x. We are then interested only in the conditional probability distribution over responses $P(Y|x)$. In the rest of this work, we

will always assume the presence of such an input x, *e.g.*, an instruction or dialogue history.

2.1 Decision Rules

In order to obtain a generation from a trained language model P given some input x, it is necessary to decide on a single "best" outcome in Σ^*. Formally, this requires a decision rule that defines a mapping from a distribution P to such an outcome y^*. A common choice is to output the highest probability outcome under $P(Y|x)$, a decision rule known as *maximum-a-posteriori*, typically approximated using beam search or majority voting.

$$y^*_{\text{MAP}} = \underset{h \in \Sigma^*}{\text{argmax}}\, P(Y = h|x) \tag{2}$$

However, studies have shown that model probability does not reliably align with human preferences (Stahlberg and Byrne, 2019; Zhang et al., 2021), and Minimum Bayes Risk (MBR) has become a popular alternative. MBR stems from the principle of maximisation of expected utility (Berger, 1985). It requires choosing a *utility* function $u(h, r)$ that measures the benefit of choosing hypothesis h given an ideal decision r. In natural language generation, u is typically chosen to be a strong sentence similarity metric such as BLEURT (Sellam et al., 2020; Freitag et al., 2022), COMET (Rei et al., 2020; Fernandes et al., 2022) or BERTScore (Zhang et al., 2020; Suzgun et al., 2023). MBR then selects the outcome maximising utility in expectation under the model distribution:

$$y^*_{\text{MBR}} = \underset{h \in \Sigma^*}{\text{argmax}} \underset{P(Y|x)}{\mathbb{E}} [u(h, Y)] \tag{3}$$

Recently, a sampling-based approximation to MBR has become popular that approximates expected utility by obtaining a set of unbiased samples from the model and re-ranking them using Monte Carlo estimates of their expected utility (Eikema and Aziz, 2020, 2022). In this work, we will focus on this sampling-based approximation.

2.2 Structural Variation in Language Models

The importance of modelling uncertainty in natural language generation systems has received growing attention in recent years (Baan et al., 2023). Crucially, uncertainty extends beyond surface-form variations in outcome space to encompass deeper variation in latent space. To capture such variation, metrics like semantic entropy (Kuhn et al., 2024) and similarity-sensitive entropy (Cheng and Vlachos, 2024) have been proposed, primarily to identify when high uncertainty may signal potential model errors. Complementary work has examined similar measures with a different aim: to assess whether the uncertainty exhibited by language models aligns with the natural variability found in human-generated responses (Deng et al., 2022; Giulianelli et al., 2023; Ilia and Aziz, 2024).

Recent applications of MBR have largely focused on neural machine translation—a relatively constrained task where, nonetheless, models have been shown to capture less variation than what human annotators consider plausible (Giulianelli et al., 2023). Extending beyond translation, a few studies have applied MBR to other generation tasks. For example, Suzgun et al. (2023) successfully use BERTScore-based MBR for summarisation, data-to-text generation, textual style transfer, and image captioning. However, these tasks also tend to involve a limited range of plausible outputs. More recently, Wu et al. (2025) applied MBR to instruction-following tasks using an LLM-as-a-judge as a utility function. While this method yields strong results, it relies on a distillation step to approximate the utility, as directly querying an LLM judge during decoding is computationally prohibitive. In this work, we propose three lightweight adaptations to standard similarity-based utility functions, specifically designed for open-ended tasks characterised by high variability in latent structure.

3 Structure-Conditional Optimality

The central question addressed in this paper is how commonly employed utility functions for MBR decoding behave when complex structural variation is present. In Fig. 1, we illustrate the problem with a simplified example to highlight how the choice of utility function can influence decision-making—particularly when the outcome space contains multiple distinct modes. In this example, the outcome space is modelled as a bimodal Gaussian, and the decision problem is to select a single "best" outcome on the real line. If we use the negative squared error as our utility function,[1] the theoretical optimum corresponds to the mean of the bimodal distribution (the light blue line in Fig. 1). This solution may be undesirable as the mean lies in a region of low probability mass and is unlikely to be sampled in practice. If we apply a sampling-

[1] Equivalently, one may frame this as minimising the risk under a squared error loss function.

based approximation to the decision rule, as is common in language generation applications of MBR, the approximation selects an outcome near this theoretical optimum, which typically resides at the boundary of one of the clusters. Alternatively, if we adopt a different utility function—such as a radial basis function kernel—the theoretical optimum shifts to the mode of the largest cluster (Fig. 1, dark blue line). This outcome, being more representative of a high-probability region, may be more desirable than either the low-probability intermodal mean or an outcome near the edge of a cluster.

In probability distributions over natural language, multiple such "modes" may also be present, albeit more difficult to define and detect. Generations might cluster around various semantically distinct plausible answers to a question, differing intended dialogue acts in a response, or varying response structure. Depending on the utility function used, this can result in behaviour analogous to that shown in Fig. 1. Whether this is desirable depends on the decision-maker; for instance, it may be appropriate if the model assigns probability mass to responses like *The answer could be either [A] or [B]*, but in other cases, it could lead to suboptimal decisions. In this work, we investigate this phenomenon and propose simple adaptations to utility functions that encourage behaviour more similar to that of the RBF utility in the continuous example.

3.1 Evaluating Structural Sensitivity in MBR

To quantify the extent to which the MBR solution with commonly used utility functions respects structural variability in outcome spaces over natural language, we introduce two complementary metrics. These metrics evaluate whether MBR solutions align with, or differ from, solutions obtained when conditioning on latent structures.

Cluster Optimality. This metric quantifies the proportion of cases, over a test set, in which the MBR solution under the distribution $P(Y|x)$ matches the MBR solution under the conditional distribution $P(Y|x, s)$, where s denotes an annotated structure (*e.g.*, a dialogue act) that we additionally condition on. Formally, let

$$\hat{y}_i = \operatorname*{argmax}_h \mathop{\mathbb{E}}_{P(Y|x_i)}[u(h, Y)] \tag{4}$$

be the MBR solution for input i, and

$$\hat{y}_i^{(s)} = \operatorname*{argmax}_h \mathop{\mathbb{E}}_{P(Y|x_i,s)}[u(h, Y)] \tag{5}$$

the MBR solution conditioned on s. The cluster optimality metric is then defined, for test set $\mathcal{D}$, as

$$\text{CO} = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \mathbb{1}\{\hat{y}_i = \hat{y}_i^{(s)}\} \tag{6}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function.

Cluster-Optimal Rank Correlation. In addition to the top-ranked solution, we also examine the full rankings produced by MBR. For each input i, consider a fixed set of hypothesis generations $\mathcal{H}_i = \{h_1, \ldots, h_n\}$ corresponding to structure s. Define the rankings:

$$R_{ij} = \text{rank of } h_j \text{ by } \mathop{\mathbb{E}}_{P(Y|x_i)}[u(h_j, Y)] \tag{7}$$

$$R_{ij}^{(s)} = \text{rank of } h_j \text{ by } \mathop{\mathbb{E}}_{P(Y|x_i,s)}[u(h_j, Y)] \tag{8}$$

The cluster-optimal rank correlation is then the average Spearman's rank correlation coefficient ρ between these two rankings over the test set:

$$\text{CORC} = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \rho\big(R_i, R_i^{(s)}\big) \tag{9}$$

4 Standard Utility Functions are Not Structure-Conditionally Optimal

We now demonstrate that MBR solutions derived using standard utility functions, such as BERTScore or BLEURT, often diverge from those obtained when conditioning on latent structural representations. While this divergence may be acceptable from the perspective of the decision-maker, our analysis adopts a different premise. We assume a production process in which the speaker first selects a latent structure—implicitly or explicitly—and then realises it through an utterance. Under this assumption, a generation should be optimal with respect to some latent structure, specifically the one selected by the speaker. Note that we do not model the initial stage of this process, *i.e.*, the selection or planning of the latent structure. Instead, we take it as given and focus on the requirement that the resulting generation be optimal within plausible realisations of the chosen structure.

To investigate how sensitive the MBR solution is to structural uncertainty in the outcome space, we consider three representative types of latent structure—dialogue act, emotion, and response structure—each of these defines a plausible axis of variation in generated text (see §4.1). For each

structure, we construct a dataset that reflects the outcome space of a hypothetical model with uncertainty over its possible instantiations. We then compute the standard MBR solution over the entire space, and assess its optimality using the evaluation criteria introduced in §3.1. The results of this analysis are summarised in Tab. 1 and presented in §4.2.

4.1 Constructing Outcome Spaces with Controlled Structural Uncertainty

We ground our analysis in three types of latent structure. This section introduces each structure type and describes how we construct datasets to model uncertainty over their possible instantiations.

4.1.1 Latent Structure Types

We examine three types of latent structure that are representative of structural variability in the outcome spaces of open-ended generation tasks.

Dialogue Act. A dialogue act represents the communicative function or intent of an utterance within the context of a conversation. Following the taxonomy proposed by Amanova et al. (2016), we focus on four dialogue act types: INFORM, QUESTION, COMMISSIVE, and DIRECTIVE. Each of the four example utterances presented in the introduction exemplifies one of these categories.

Emotion. Another latent factor that shapes the form of an utterance in conversation is the emotion the speaker aims to express. In this work, we adopt Ekman's six basic emotions (Ekman, 1992): HAPPINESS, SADNESS, FEAR, ANGER, SURPRISE, and DISGUST. These emotional states influence both lexical choice and broader stylistic features.

Response Structure. This structure type captures how information is organised within an instruction-following response. We consider four ad-hoc categories: BRIEF, a single-sentence reply; PARAGRAPH, a more developed, single-paragraph answer; LIST, a bullet-pointed set of items; and TABLE, a structured tabular presentation.

4.1.2 Dataset Construction

For each type of latent structure, we construct a dataset that simulates the outcome space of a hypothetical model with uncertainty over possible instantiations of that structure. As generation contexts, we randomly sample conversational contexts from the DailyDialog corpus (Li et al., 2017)—1,000 each for dialogue act and emotion—and take the first 1,000 instructions from the Alpaca dataset (Taori et al., 2023) for response structure. We then prompt the instruction-tuned, 13 billion parameter variant[2] of the OLMo 2 model suite (OLMo et al., 2025) to generate outputs for each category within each structure type, using hand-curated prompts (see App. A for details). For every context, we generate 25 responses per structure category (*e.g.*, 25 BRIEF, 25 PARAGRAPH, 25 LIST, and 25 TABLE responses). This procedure results in 3,000 distinct outcome spaces, corresponding to 350,000 candidate generations in total.[3]

Metric	Utility	Dial. Act	Emotion	Resp. Str.	All (Avg)
CO	BERTScore	0.370	0.330	0.390	0.363
	BLEURT	0.410	0.510	0.530	0.483
CORC	BERTScore	0.081	0.084	0.080	0.082
	BLEURT	0.144	0.155	0.123	0.141

Table 1: Cluster Optimality (CO) and Cluster-Optimal Rank Correlation (CORC) of MBR solutions obtained using BERTScore and BLEURT utility functions over constructed outcome spaces.

4.2 Structural Sensitivity of Standard MBR Utility Functions

Tab. 1 presents cluster optimality (CO, Eq. 6) and cluster-optimal rank correlation (CORC, Eq. 9) scores for MBR solutions under two standard utility functions across our three latent structure types. These metrics quantify how often the MBR-selected response is optimal with respect to the latent structure (CO), and how well it aligns with the structure-optimal ranking (CORC).

Across all structure types, we observe a consistent degree of suboptimality. The CO scores indicate that in fewer than half of the cases, the MBR solution is optimal with respect to its underlying structure (36.3% using BERTScore, 48.3% with BLEURT). This misalignment persists across dialogue act, emotion, and response structure, with no evident correlation to the number of clusters involved, suggesting that the failure to recover structure-optimal responses is not merely due to increased structural granularity. While slight differences are present between BLEURT and BERTScore, both utility functions consistently select suboptimal generations and demonstrate relatively weak ranking correlation. Overall, this analysis shows that standard utility functions possess low sensitivity to structural uncertainty.

[2] `allenai/OLMo-2-1124-13B-Instruct`

[3] The full dataset is included as Supplementary Material and will be released publicly upon acceptance.

5 Structure-Conditional MBR Decoding

To address the limitations of MBR decoding with standard utility functions in the presence of structural variability, we propose three structure-aware decoding approaches.

Utility Cut-off. Standard utility functions may implicitly penalise structural mismatches, but they do not prevent structurally dissimilar candidates from influencing the ranking of outputs. To mitigate this, we introduce a simple utility cut-off mechanism that filters out low-utility comparisons when computing expected utility. Specifically, we modify the utility function $u(y, y')$ as follows:

$$u_{\text{cut}}(y, y') = \begin{cases} u(y, y') & \text{if } u(y, y') \geq \tau, \\ \delta & \text{otherwise} \end{cases} \quad (10)$$

where τ is a threshold fixed across the dataset, and δ is a small constant (or zero). This limits the influence of distant or structurally irrelevant samples, aligning the MBR solution more closely with local modes in the outcome distribution.

Clustering. A more explicit approach to structure-aware decoding is to first partition the outcome space into clusters—each corresponding to a distinct latent structure—and then apply MBR within the dominant cluster. We implement this by clustering candidate generations using sequence embeddings $\phi(y)$ derived from a model fine-tuned to detect particular structures of interest (*e.g.*, dialogue act, response structure, or affective content). Formally, let $\mathcal{H} = \{h_1, \ldots, h_n\}$ be the set of candidates, and let $C_1, \ldots, C_k$ denote the resulting clusters, with $\mathcal{H} = \bigcup_{j=1}^{k} C_j$. At inference time, we restrict MBR decoding to the members of the largest cluster $C^\star = \text{argmax}_{C_j} |C_j|$ such that

$$\hat{y} = \underset{h \in C^\star}{\text{argmax}} \underset{P(Y|x)}{\mathbb{E}} [u(h, Y) \mid Y \in C^\star]. \quad (11)$$

To recover a full ranking over candidates (*e.g.*, for evaluation), we first rank clusters by size, and then rank candidates within each cluster based on expected utility. This two-stage process prioritises high-utility responses as judged against structurally consistent pseudo-references, reducing the risk of inter-modal averaging in the selected outputs.

This procedure could also theoretically be formulated as an adaptation of the utility function:

$$u_{\text{cl}}(y, y') = \mathbb{1}\{C(y) = C(y')\} \times \\ \times u(y, y') \times \mathbb{1}\{C(y) = C^*\}, \quad (12)$$

where C^* represents the cluster with highest probability mass under $P(Y|x)$. Decoding then becomes standard MBR maximisation of expected utility under the adapted utility function.

Structure Embeddings. As an alternative to explicit clustering, we propose incorporating structural sensitivity into the utility function by leveraging structure-aware sequence embeddings. Specifically, we fine-tune a sequence embedding model to encode the structural property of interest and redefine the utility function to weight candidate comparisons by their similarity in this embedding space. Formally, for a candidate y and a reference y', we compute the modified utility as:

$$u_{\text{emb}}(y, y') = u(y, y') \cdot \cos\big(\phi(y), \phi(y')\big), \quad (13)$$

where $u(y, y')$ is the original utility and $\cos(\cdot)$ denotes the cosine similarity between structure-sensitive embeddings ϕ. To further reduce the influence of structurally mismatched samples, we also experiment with a threshold on cosine similarity: values below the threshold are set to zero, removing the contribution of the utility comparison to the expected utility altogether. This approach allows us to softly bias the MBR solution toward structurally coherent outputs, without requiring hard clustering or explicit structure labels at inference time.

6 Experiments

To evaluate the effectiveness of the proposed methods, we conduct a series of experiments on the dataset we constructed in §4, as well as two real-world instruction-following datasets. All our experiments use either BERTScore or BLEURT as the base utility function, two commonly employed utility functions in natural language generation (Freitag et al., 2022; Suzgun et al., 2023).

6.1 Cluster Optimality Under Controlled Structural Uncertainty

We first assess our methods on the three datasets constructed in §4, which contain generations consisting of various types of structural uncertainty: over dialogue acts, emotions, and response structures. Recall that we treat these generations as hypothetical outcome spaces of a language model. That is, we consider all generations for a given context to be unbiased samples from a language model that we wish to perform MBR decoding with. We split the 1,000 contexts in each dataset into training, validation, and test sets using an 800/100/100 split.

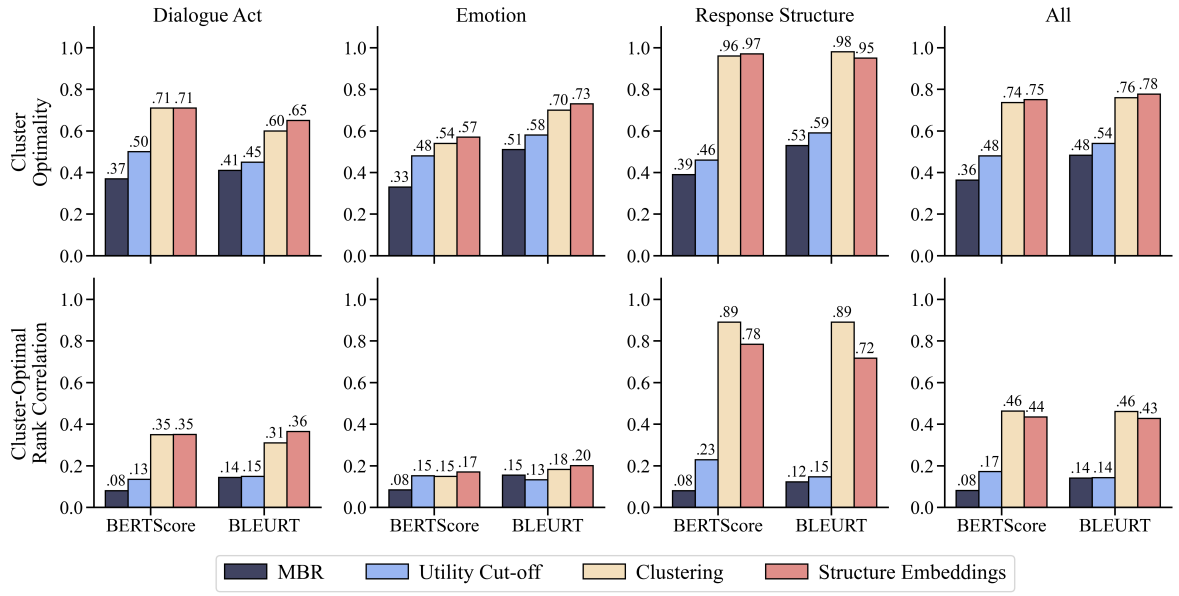

Figure 2: Cluster Optimality and Cluster-Optimal Rank Correlation on the constructed outcome spaces of §4.1. We compare standard BERTScore and BLEURT MBR with the three adaptations to the utility functions proposed in §5.

Hyperparameter Selection. For each method proposed in §5, we use the training and validation splits to select hyperparameters and train the sequence embedding models. Determining the threshold in the Utility Cut-off approach is conducted separately for BERTScore and BLEURT, resulting in different thresholds. We base our sequence embedding models on the `all-mpnet-base-v2`[4] Sentence Transformer (Reimers and Gurevych, 2019), which we further fine-tuned using a triplet loss and gold annotations of underlying structure to enhance sensitivity to the structural variation present in our datasets. We use the same sequence embedding models for our Clustering and Structure Embeddings approaches. We find that jointly fine-tuning and selecting thresholds on the combination of all three types of latent structure leads to the most robust performance in terms of CO,[5] and we use these models for the experiments below. Further details on the hyperparameter selection and fine-tuning procedures can be found in App. B.

Results. We compare each of our proposed methods against standard sampling-based MBR decoding using either BERTScore or BLEURT as the utility function, and measure both cluster optimality (CO, Eq. 6) and cluster optimal ranking correlation (CORC, Eq. 9). Results are shown in Fig. 2. All the methods we proposed improve cluster optimality compared to the baseline versions of BERTScore and BLEURT. Utility Cut-off yields the smallest improvement over standard BERTScore and BLEURT MBR, on average increasing CO by 11.7% and 5.6%, respectively, and CORC by 0.091 and 0.002, respectively. The Clustering and Structure Embeddings approaches perform considerably better than baseline MBR. Clustering improves CO on average by 37.3%/27.7% and CORC by 0.382/0.320 CORC over standard BERTScore and BLEURT MBR, respectively. Similarly, Structure Embeddings improve CO on average by 38.7%/29.3% CO and CORC by 0.354/0.287 over standard BERTScore and BLEURT MBR, respectively. We note again that higher CO does not always correspond with higher CORC, indicating that achieving the cluster-optimal MBR solution is generally easier than recovering the entire ranking accurately. Additionally, we observe that some types of latent structure are more difficult to capture effectively than others.

6.2 Instruction-Following

Next, we evaluate our methods on two real-world instruction-following datasets: AlpacaEval (Li et al., 2023) and MT-Bench (Zheng et al., 2023). In this case, we do not have access to any labelling of potential latent structure. We use the same hyperparameters and sequence embedding models from the previous set of experiments, tuned on the combination of all three datasets from §4.

[4] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[5] Generally, we find CO and CORC in validation procedures to align reasonably well.

Benchmark	MBR	Cut-off	Cluster	Embeddings
AlpacaEval	96.5%	96.1%	**97.0%**	96.1%
MT-Bench (single)	76.3%	**90.0%**	80.0%	78.8%
MT-Bench (multi)	71.3%	70.0%	72.5%	**74.4%**

Table 2: AlpacaEval and MT-Bench Prometheus win rates versus `text-davinci-003` (AlpacaEval) / GPT-4o (MT-Bench). We compare standard BERTScore MBR with the approaches introduced in §5: Utility **Cut-off**, **Cluster**ing and Structure **Embeddings**.

As a language model, we select OLMo 2 (13B) (OLMo et al., 2025), and obtain 30 unbiased samples per prompt for use in MBR decoding. To measure task performance, we use Prometheus[6] (Kim et al., 2024) as a judge, conducting relative grading against `text-davinci-003` and GPT-4o (OpenAI, 2024) for AlpacaEval and MT-Bench, respectively.[7] All experiments employ BERTScore as the base utility. Further details on the generation and evaluation procedures are provided in App. C.

Results. Tab. 2 reports win rates against `text-davinci-003` and GPT-4o for standard MBR decoding with a BERTScore utility, alongside our structure-conditional utilities from §5. On AlpacaEval, the Clustering method outperforms standard BERTScore MBR. In the single-turn MT-Bench setting, both Clustering and Utility Cut-off surpass standard MBR, with Utility Cut-off achieving a notable 13.7 percentage point improvement and reaching a 90% win rate over GPT-4o. This indicates responses are often judged clearer, more helpful, accurate, and fully aligned with the intended purpose of the instruction. Performance declines across the board in the more challenging multi-turn MT-Bench setting. However, both clustering and structure embeddings still outperform standard MBR, demonstrating improved structural sensitivity also in extended interactions. Smaller gains here may stem from reduced uncertainty as conversational context accumulates, resulting in less diverse outcome spaces. In such cases, structure-conditional utilities likely yield results similar to standard MBR, reducing the relative benefit of structural adaptations. We also observe that structure embeddings tend to outperform clustering, possibly because soft partitioning better captures subtle structural differences, whereas hard clustering might inadvertently exclude partially similar candidates. Nonetheless, the overall lower MBR performance in multi-turn tasks indicates these settings are inherently more challenging, beyond just reduced variability. Overall, the consistent improvements of structure-aware MBR methods over standard MBR suggest that incorporating latent structural information not only enhances the theoretical optimality of MBR solutions but also improves generation quality in practical settings.

[6] `prometheus-eval/prometheus-7b-v2.0`

[7] We did not find any available multi-turn system generations for the full MT-Bench dataset. Therefore, we generated our own from OpenAI's GPT-4o using greedy decoding.

7 Conclusion

In this work, we examined the limitations of MBR decoding in open-ended generation scenarios, where outcome spaces might exhibit high structural variability. We hypothesised that commonly used utility functions are insufficiently sensitive to latent structural uncertainty, leading to suboptimal generation choices within structurally coherent clusters of responses. To evaluate this hypothesis, we constructed a dataset featuring naturally occurring contexts paired with outcome spaces that exhibit controlled variation in dialogue act, emotion, and response structure. Our findings confirm that MBR decoding under standard utilities frequently fails to select cluster-optimal candidates, with suboptimal selections occurring in more than half of the cases.

To address this issue, we proposed three approaches to adapt utility functions to be more structurally aware. The corresponding approaches—Clustering, Structure Embeddings, and Utility Cut-off—demonstrate significant improvements in both cluster optimality (up to 98% for response structure) and cluster-optimal rank correlation (up to 0.89 for response structure). Importantly, these methods incur only modest additional computational cost, requiring only lightweight fine-tuning of a sequence embedding model or performing a hyperparameter search for a threshold value.

Based on these results, we recommend adopting structure-aware MBR decoding in tasks characterised by medium to high outcome space variability, such as instruction-following and conversational tasks. We encourage future research into the development of structure-sensitive utility functions that build on this work to achieve even greater cluster optimality, generation quality, or inference-time efficiency. In addition, we see value in further investigating the relationship between outcome space variability and the effectiveness of structure-aware MBR, as well as the connection between cluster optimality and overall generation quality.

Limitations

To test our hypothesis on the sub-optimality of standard similarity-based MBR utility functions, we relied on a curated dataset that captures three representative types of latent structure commonly found in open-ended natural language generation tasks. However, this dataset does not exhaustively cover all possible structural variations present in natural language. Additionally, our evaluation assumes that language models accurately represent uncertainty over latent structures—an assumption that may not always hold in practice (see, e.g., Giulianelli et al., 2023). For example, in a dialogue setting, a model might assign most of its probability mass to responses aligned with the INFORM category, even if human responses display a broader range of structural types.

In terms of computational requirements, our methods introduce minimal overhead beyond standard MBR decoding. However, it is important to note that MBR decoding itself *is* significantly more computationally demanding than greedy decoding or sampling a single generation. Since our approaches build on MBR, they inherit this higher computational cost. That said, we believe our methods stand to benefit from recent advances aimed at improving the efficiency of MBR decoding (Cheng and Vlachos, 2023; Vamvas and Sennrich, 2024; Yang et al., 2024).

Finally, in our evaluation on instruction-following datasets, we rely on an LLM-as-a-judge: Prometheus (Kim et al., 2024). These are imperfect evaluators, may be biased towards particular types of responses (Wang et al., 2024; Stureborg et al., 2024), such as more elaborate ones, and do not always align with human judgements (Zeng et al., 2024; Bavaresco et al., 2024). Additionally, Prometheus relies on a predefined rubric, and its performance may be sensitive to the specific formulation of that rubric. We did not conduct extensive experiments with alternative rubric designs, which may influence the robustness of the results.

References

Dilafruz Amanova, Volha Petukhova, and Dietrich Klakow. 2016. Creating annotated dialogue resources: Cross-domain dialogue act classification. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 111–117, Portorož, Slovenia. European Language Resources Association (ELRA).

Joris Baan, Nico Daheim, Evgenia Ilia, Dennis Ulmer, Haau-Sing Li, Raquel Fernández, Barbara Plank, Rico Sennrich, Chrysoula Zerva, and Wilker Aziz. 2023. Uncertainty in natural language generation: From theory to applications.

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, André F. T. Martins, Philipp Mondorf, Vera Neplenbroek, Sandro Pezzelle, Barbara Plank, David Schlangen, Alessandro Suglia, Aditya K Surikuchi, Ece Takmaz, and Alberto Testoni. 2024. LLMs instead of human judges? A large scale empirical study across 20 NLP evaluation tasks.

James O Berger. 1985. *Statistical decision theory and Bayesian analysis; 2nd ed.* Springer Series in Statistics. Springer, New York.

Julius Cheng and Andreas Vlachos. 2023. Faster minimum Bayes risk decoding with confidence-based pruning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12473–12480, Singapore. Association for Computational Linguistics.

Julius Cheng and Andreas Vlachos. 2024. Measuring uncertainty in neural machine translation with similarity-sensitive entropy. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2115–2128, St. Julian's, Malta. Association for Computational Linguistics.

Yuntian Deng, Volodymyr Kuleshov, and Alexander Rush. 2022. Model criticism for long-form text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11887–11912, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Bryan Eikema and Wilker Aziz. 2020. Is MAP decoding all you need? the inadequacy of the mode in neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4506–4520, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Bryan Eikema and Wilker Aziz. 2022. Sampling-based approximations to minimum Bayes risk decoding for neural machine translation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10978–10993, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

P Ekman. 1992. Are there basic emotions? *Psychological review*, 99(3):550—553.

Patrick Fernandes, António Farinhas, Ricardo Rei, José G. C. de Souza, Perez Ogayo, Graham Neubig, and Andre Martins. 2022. Quality-aware decoding for neural machine translation. In *Proceedings of*

the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1396–1412, Seattle, United States. Association for Computational Linguistics.

Markus Freitag, David Grangier, Qijun Tan, and Bowen Liang. 2022. High quality rather than high model probability: Minimum Bayes risk decoding with neural metrics. *Transactions of the Association for Computational Linguistics*, 10:811–825.

Mario Giulianelli, Joris Baan, Wilker Aziz, Raquel Fernández, and Barbara Plank. 2023. What comes next? evaluating uncertainty in neural text generators against human production variability. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14349–14371, Singapore. Association for Computational Linguistics.

Evgenia Ilia and Wilker Aziz. 2024. Variability need not imply error: The case of adequate but semantically distinct responses.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2024. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*.

Shankar Kumar and William Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 169–176, Boston, Massachusetts, USA. Association for Computational Linguistics.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of The 8th International Joint Conference on Natural Language Processing (IJCNLP 2017)*.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2025. 2 olmo 2 furious.

OpenAI. 2024. Gpt-4o system card.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

Rickard Stureborg, Dimitris Alikaniotis, and Yoshi Suhara. 2024. Large language models are inconsistent and biased evaluators.

Mirac Suzgun, Luke Melas-Kyriazi, and Dan Jurafsky. 2023. Follow the wisdom of the crowd: Effective text generation via minimum Bayes risk decoding. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4265–4293, Toronto, Canada. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Jannis Vamvas and Rico Sennrich. 2024. Linear-time minimum Bayes risk decoding with reference aggregation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*

(Volume 2: Short Papers), pages 790–801, Bangkok, Thailand. Association for Computational Linguistics.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. 2024. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9440–9450, Bangkok, Thailand. Association for Computational Linguistics.

Ian Wu, Patrick Fernandes, Amanda Bertsch, Seungone Kim, Sina Khoshfetrat Pakazad, and Graham Neubig. 2025. Better instruction-following through minimum bayes risk. In *The Thirteenth International Conference on Learning Representations*.

Guangyu Yang, Jinghong Chen, Weizhe Lin, and Bill Byrne. 2024. Direct preference optimization for neural machine translation with minimum Bayes risk decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 391–398, Mexico City, Mexico. Association for Computational Linguistics.

Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. 2024. Evaluating large language models at evaluating instruction following. In *The Twelfth International Conference on Learning Representations*.

Hugh Zhang, Daniel Duckworth, Daphne Ippolito, and Arvind Neelakantan. 2021. Trading off diversity and quality in natural language generation. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 25–33, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena.

A Data Generation

In §4.1.2 we semi-automatically construct a natural language dataset of hypothetical outcome spaces with varying underlying latent structure: dialogue act, emotion and response structure. Here, we describe the full generation procedure in more detail.

A.1 Generation Procedure

We use the input prompts from the DailyDialog dataset (Li et al., 2017) as a basis for the dialogue act and emotion subsets, and Alpaca (Taori et al., 2023) as a basis for the response structure subset. In order to obtain sampled generations, we use `allenai/OLMo-2-1124-13B-Instruct` (OLMo et al., 2025).

1. **Pre-processing.** For DailyDialog, we remove unnecessary white spaces, and filter dialogues by the number of turns (only dialogues with ≥ 2 turns are kept). For each of the dialogues, we also randomly sample the number of turns which we will use from the dialogue, between two and the total number of available turns minus two. We also prepend *"A:"* or *"B:"* prefixes to each turn to indicate the speaker. For Alpaca, we only filter out any prompts that have an additional `input` field.

2. **Prompt creation**. We randomly select 1,000 dialogues twice from the DailyDialog dataset and the first 1,000 instructions without `input` field from the Alpaca dataset. We then fill out the predefined prompt templates (defined in App. A.2) with the selected examples, resulting in prompts for each pre-defined category within every latent structure. In total, we create $1,000 * 4 = 4,000$ inputs for the dialogue act subset, $1,000 * 4 = 4,000$ inputs for response structure subset, and $1,000 * 6 = 6,000$ inputs for the emotion subset to be fed into OLMo.

3. **Generation**. Using OLMo, we then generate 25 unbiased samples for each of the $14,000$ constructed input prompts.

4. **Post-processing**. We finalize the procedure by removing quotation marks around the generation and stripping any *"A:"* or *"B:"* prefixes at the start of generations.

A.2 Prompt Templates

To generate prompts covering all categories within a latent structure, we define three types of prompt templates - one per latent structure.

A.2.1 Dialogue Act

For the dialogue act subset, we define a `responder` as the speaker whose turn it is now to speak (*e.g.*if the dialogue excerpt ends at `A`'s turn, we define `responder = B`). For each `dialogue from DailyDialog` we then iterate through the four defined dialogue acts and pass each of them as `act_name`.

```
### **Types of Dialogue Acts**
Here are common categories of dialogue
    acts, though exact categorizations
    may vary depending on the framework:
#### **1. Inform**
  - The Inform class contains all
      statements and questions by which
      the speaker shares information
      with the listener. The speaker
      assumes the information is
      correct and believes the
      addressee does not know or is not
      aware of it yet.
  - **Examples**:
    - "The meeting starts at 3 PM."
    - "I've already emailed the report."
    - "I saw John at the store
        yesterday."
    - "William Shakespeare wrote it."
    - "The Eiffel Tower is in Paris."

#### **2. Question**
  - The Question class includes speech
      acts where the speaker seeks
      information by asking a question.
      These acts are used when the
      speaker wants to know something
      and believes the listener has the
      answer. Questions can take
      different forms, including
      Propositional Questions (yes/no
      questions), Check Questions
      (confirming known information),
      Set Questions (open-ended
      questions), and Choice Questions
      (questions with multiple options).
  - **Examples**:
    - "Did you finish your assignment?"
    - "You've met Sarah before, haven't
        you?"
    - "What time does the meeting
        start?"
    - "Could you clarify what you meant
        by that?"
    - "Do you prefer coffee or tea?"

#### **3. Directive**
  - The Directive class includes speech
      acts where the speaker wants the
      listener to perform an action.
      This class covers Requests
      (asking someone to do something),
      Instructions (giving direct
      orders or guidance), Suggestions
      (offering recommendations), and
      Accepting or Rejecting Offers
      (responding to proposals). These
      acts differ based on how much
      pressure the speaker applies and
      their assumptions about the
      listener's willingness and
      ability to comply.
  - **Examples**:
    - "Can you send me the file?"
    - "Fill out this form before the
        appointment."
    - "You should try the new Italian
        restaurant downtown."
    - "Yes, I'd love to join you for
        dinner!"
    - "No, I can't take on another
        project right now."

#### **4. Commissive**
  - The Commissive class involves
      speech acts where the speaker
      commits to performing an action
      in the future. These acts include
      Accepting or Rejecting Requests,
      Suggestions, and Offers. By
      performing a Commissive act, the
      speaker is making a promise or
      commitment to carry out the
      action requested, suggested, or
      offered. These acts reflect the
      speaker's willingness to take
      responsibility for fulfilling the
      commitment, whether by agreeing
      to a proposal or refusing it.
  - **Examples**:
    - "Fine, I'll pick you up at 5 PM."
    - "Sorry, I can't do that right
        now."
    - "That sounds great, I'll take the
        promotion."
    - "I promise to finish the report
        by the end of the day."
    - "I'll make sure to take care of
        it this weekend."

---

### **Dialogue Excerpt**

{dialogue from DailyDialog}

---

### **Instructions**
Please consider the provided dialogue
    excerpt and provide a plausible
    response (and only a single
    response) for {responder} that
    reflects the following dialogue
    act: {act_name}. Output only
    {responder}'s response with no
    additional text.<end_of_prompt>
```

A.2.2 Emotion

For the emotion subset, we again define a `responder` in the same way as in App. A.2.1. For each `dialogue from DailyDialog` we then iterate through the six defined emotions and pass each of them as `emotion_name`.

```
   ### **Types of Emotions**
Here are seven main categories of
    emotions.

#### **1. Anger**
  - The Anger category represents
      emotions related to feelings of
      displeasure, hostility, or
      frustration. This emotion often
      arises when someone feels wronged
      or blocked from achieving their
```

```
      goal. It can range from mild
      irritation to intense rage.
  - **Examples**:
    - "I can't believe this is
      happening!"
    - "This is so unfair!"
    - "Why does everything always go
      wrong for me?"
    - "I'm so frustrated with this
      situation!"
    - "I'm really mad about how things
      turned out."

#### **2. Disgust**
  - The Disgust category includes
    emotions related to a strong
    sense of revulsion, disapproval,
    or distaste. It often arises when
    something is perceived as
    offensive, repellent, or morally
    objectionable.
  - **Examples**:
    - "That food looks awful!"
    - "I can't stand how they treat
      people."
    - "This is disgusting. I can't
      believe they did that."
    - "I feel sick just thinking about
      it."
    - "That's absolutely revolting!"

#### **3. Fear**
  - The Fear category includes emotions
    related to anxiety, nervousness,
    and concern about possible danger
    or harm. Fear can be rational or
    irrational and may cause physical
    or psychological distress.
  - **Examples**:
    - "I'm really scared about what's
      going to happen."
    - "I don't know if I can handle
      this situation."
    - "What if things don't go as
      planned?"
    - "I'm afraid something bad might
      happen."
    - "I'm nervous about the meeting
      this morning."

#### **4. Happiness**
  - The Happiness category includes
    emotions related to joy,
    contentment, and pleasure.
    Happiness is often associated
    with positive experiences,
    accomplishments, and satisfying
    events.
  - **Examples**:
    - "I'm so excited about this
      weekend!"
    - "This is such a great day!"
    - "I feel so happy about my
      progress."
    - "That sounds amazing, I'm really
      looking forward to it!"
    - "I'm so glad everything worked
      out!"

#### **5. Sadness**
  - The Sadness category represents
    emotions related to feelings of
    loss, disappointment, or sorrow.
    It often arises when there is a
    sense of unmet expectations,
    failure, or grief.
  - **Examples**:
    - "I feel so down about what
      happened."
    - "I can't stop thinking about it,
      it's just so upsetting."
    - "I'm really sad things turned out
      this way."
    - "It's been a tough time, and I
      feel heartbroken."
    - "I don't know how to get over
      this sadness."

#### **6. Surprise**
  - The Surprise category represents
    emotions related to unexpected
    events or outcomes, ranging from
    shock to awe. This emotion can be
    positive or negative, depending
    on the nature of the surprise.
  - **Examples**:
    - "Wow, I didn't see that coming!"
    - "That's such a surprise, I can't
      believe it!"
    - "I'm totally shocked by what
      happened."
    - "I wasn't expecting that at all!"
    - "I'm so surprised you did that!"

---

### **Dialogue Excerpt**

{dialogue from DailyDialog}

---

### **Instructions**
Please consider the provided dialogue
    excerpt and provide a plausible
    response (and only a single
    response) for {responder} that
    reflects the following emotion:
    {emotion_name}. Output only
    {responder}'s response with no
    additional text.<end_of_prompt>
```

A.2.3 Response Structure

For the response structure subset, we define four different prompt templates, one per category of response structure. For each `prompt from Alpaca` we then append each of these templates, resulting in four different prompts, one per category, per input instruction.

BRIEF

```
{prompt from Alpaca} Give me a brief
    sentence with the answer. Make
    sure to restrict your response
    to a single sentence.
```

PARAGRAPH

```
{prompt from Alpaca} Write an
    extensive paragraph on the
    topic. Restrict your answer to a
    single paragraph
```

LIST

```
{prompt from Alpaca} In your answer,
    make sure to include a bullet
    point list of items relevant to
    the topic. Keep your answer
    brief and make sure it contains
    a bullet point list.
```

TABLE

```
{prompt from Alpaca} In your answer,
    include a table relevant to the
    topic. Keep your answer brief
    and make sure it contains a
    table.
```

B Hyperparameter Selection

We randomly split our generated datasets (dialogue act, emotion and response structure) into 800/100/100 training/validation/testing data points. All data consists of multiple labelled clusters with 25 generations per cluster. We compute BERTScore and BLEURT MBR solutions conditioned on each labelled cluster to get cluster-optimal rankings and MBR solutions to compare to. We use the training and validation splits for fine-tuning sequence embedding models and for hyperparameter selection. We have performed all training and hyperparameter selection both on individual datasets (either dialogue act, emotion or response structure) as well as on the combination of all datasets. We find that models trained on all data perform best overall and thus have used those in experiments. We proceed here to discuss the results of hyperparameter selection for each individual approach in more detail.

Utility Cut-off. We considered both an absolute threshold on the utility value as well as a threshold on the deviation from the highest observed utility in the sample. We don't consider any utility comparisons with the candidate itself, *i.e.*, we mask out the diagonal of the utility matrix. Furthermore, we experiment with both setting utility values below the threshold to 0 or -1 as well as discarding those utility comparisons altogether. We test a range of 50 threshold values ranging within reasonable values for the utility function itself, and order settings based on cluster optimality on the training data. We then take the 10 best performing setups and select the one that performs best in terms of cluster optimality on the validation data. We tune the threshold independently for both BLEURT and BERTScore. We find an absolute value threshold to work superior for both utilities, as well as to zero out values below the threshold. We find an optimal threshold of 0.512 and 0.918 for BLEURT and BERTScore respectively.

Clustering. We use the Sentence Transformers `all-mpnet-base-v2` model as a basis for obtaining sequence embeddings. We further fine-tune this model using a triplet loss on triplets from our labelled datasets. We experiment with learning rates between 1×10^{-4}, 1×10^{-5} and 1×10^{-6} and find a learning rate of 1×10^{-5} to lead to best validation loss overall. We then use these sequence embeddings with the k-means algorithm to obtain clusters. We select a number of clusters based on the silhouette score for $k = [2, 6]$ and set a threshold that the silhouette scores need to reach, otherwise k is set to 1 and we consider all generations to come from a single cluster. This threshold is tuned on prediction accuracy on the number of clusters for a range of values in $(0, 1)$ on random subsamples of the validation data containing a random number of clusters per subsample.

Structure Embeddings. Here, we use the same fine-tuned sentence transformer model from the clustering approach. We shift and compress cosine similarity values to range between 0 and 1. We optionally consider a threshold on cosine similarity and perform an identical selection procedure to that for the threshold in the Utility Cut-off approach. We find that a threshold does considerably improve cluster optimality and end up with a threshold of 0.918.

Fine-Tuned Utilities: BERTScore and BLEURT. We also attempted fine-tuning BERTScore and BLEURT directly to be more sensitive to the latent structures we expect in the data. We experimented with fine-tuning BERTScore with a triplet loss on the sequence embeddings of the underlying `roberta-large` model, and used a mean squared error regression loss to fine-tune BLEURT to predict comparisons with out-of-cluster generations as 0 or -1. We attempted a range of hyperparameter values, but found that the resulting utility functions performed poorly across the board. Hence, we have not included those models in the main paper.

```
You are an advanced AI assistant
    specializing in clear,
    well-reasoned, and articulate
    responses. Your goal is to
    provide comprehensive and
    accurate answers while ensuring
    coherence, logical consistency,
    and factual correctness. Be
    precise, provide evidence-based
    explanations, and use
    structured reasoning when
    appropriate. If a question has
    multiple interpretations,
    clarify them before answering.
    Avoid unnecessary verbosity
    while maintaining completeness.
    If uncertain, state your level
    of confidence and explain why.
```

Figure 3: System prompt provided to the model for all decoding methods when generating for AlpacaEval and MT-Bench.

C Evaluation on AlpacaEval and MT-Bench

We performed evaluation on AlpacaEval and MT-Bench using Prometheus as an LLM-as-a-judge model.[8] We test the instruction-following capabilities as follows:

1. **Generation**. For each instruction from the dataset, we generate our answers from each respective decoding method. We use a system prompt given in Fig. 3. When generating for single-turn MT-Bench, we only prompt the model with the first turn and save its output. When generating for multi-turn MT-Bench, we first prompt the model with just the first turn, save its output, and then we prompt it again with both turns and the reference GPT-4o generation to the first prompt.[9] Therefore, the total number of instructions for multi-turn MT-Bench is twice the number of instructions for single-turn MT-Bench.

2. **Evaluation**. We then pass the instruction, the reference answer, as well as the generations of our decoding methods to Prometheus. The reference answers for AlpacaEval (included in the dataset) were generated by `text-davinci-003`. We collected reference answers for multi-turn MT-Bench ourselves doing greedy decoding from the GPT-4o through the OpenAI API. We use the pre-defined `RELATIVE_PROMPT_WO_REF` prompt template for Prometheus, as well as relative grading - for each pair of competing outputs, Prometheus returns one letter (*A* or *B*) defining which output is preferred. We define the rubric as *"Is the answer clear, helpful, accurate, and fully aligned with the intended purpose of the instruction?"*

3. **Final Score**. For every decoding method we then calculate that methods' win rate over the set of reference generations according to Prometheus. In multi-turn MT-Bench, we report the average of the win rates of both turns.

[8] We use `prometheus-eval/prometheus-7b-v2.0`

[9] We opted to always provide the reference response in multi-turn MT-Bench, as to not suffer from compounding errors.