LIDS: Latent Intent Decomposition for Compositional Multi-Intent SLU

Anonymous ACL submission

Abstract

Multi-intent SLU jointly models intent detection and slot filling for complex utterances. Existing methods mainly improve intent-slot interaction through graph reasoning, decoding refinement, or contrastive learning, but most approaches still rely on a single global utterance embedding, where overlapping signals may interfere during prediction. In this work, we propose LIDS, a joint model centered on latent intent decomposition. It decomposes contextual features into multiple segments, enabling different compositional patterns to contribute more independently. To further improve span representations and slot discrimination, it additionally incorporates contextual slot discrimination through contrastive learning and lightweight refinement. Experiments on MixATIS and MixSNIPS demonstrate that LIDS improves performance under challenging compositional settings. In particular, it achieves 60.1% semantic accuracy and 83.5% intent accuracy on MixATIS. Further analysis shows that different intent patterns are consistently assigned to different segments during prediction.

1 Introduction

Multi-intent SLU is commonly formulated as the joint task of intent detection and slot filling. While conventional SLU assumes a single intent per utterance, real user queries often contain multiple intents with overlapping semantics and complex intent-slot interactions (Wu et al., 2025). Existing studies mainly improve joint modeling through graph reasoning, bidirectional guidance, structured propagation, and decoding refinement (Qin et al., 2020, 2021; Xing and Tsang, 2022; Pham et al., 2023; Yin et al., 2024; Xia et al., 2025). Contrastive learning has also been introduced to improve semantic robustness and contextual consistency (Tu et al., 2023; Cheng et al., 2023; Chen et al., 2024; Zhang et al., 2025), while generative approaches reformulate SLU as conditional text generation through prompting and instruction-based frameworks (Xing et al., 2024).

Despite this progress, most existing methods still rely primarily on a single global utterance embedding for intent prediction. Under compositional settings, different intent patterns may overlap and interfere before aggregation, leading to ambiguous predictions. Existing solutions therefore improve interaction or regularize features after a global utterance embedding has already been formed, leaving the underlying interference problem insufficiently explored. Unlike conditional routing approaches that partition computation across specialized experts (Shazeer et al., 2017; Fedus et al., 2022), we instead decompose contextual features into multiple latent segments within a shared feature space. Another challenge arises from semantically similar slot labels that appear in similar contexts but correspond to different semantic roles, making decomposition and aggregation more difficult under overlapping intent structures. Hierarchical slot labels such as `fromloc.<value>` and `toloc.<value>` may contain nearly identical location expressions while indicating opposite directional roles, making them difficult to distinguish reliably.

To address these limitations, we propose LIDS, a joint model centered on Latent Intent Decomposition (LID). LID decomposes encoded features into multiple latent segments before aggregation, allowing different intent patterns to contribute more independently during prediction. LIDS further incorporates Contextual Slot Discrimination (CSD), combining contextual contrastive learning with lightweight refinement over decoded spans to stabilize decomposition-aware span representations and improve slot-role discrimination under similar contexts. Experiments on MixATIS and MixSNIPS

[0]Anonymous code and checkpoints are available at https://anonymous.4open.science/r/LIDS.

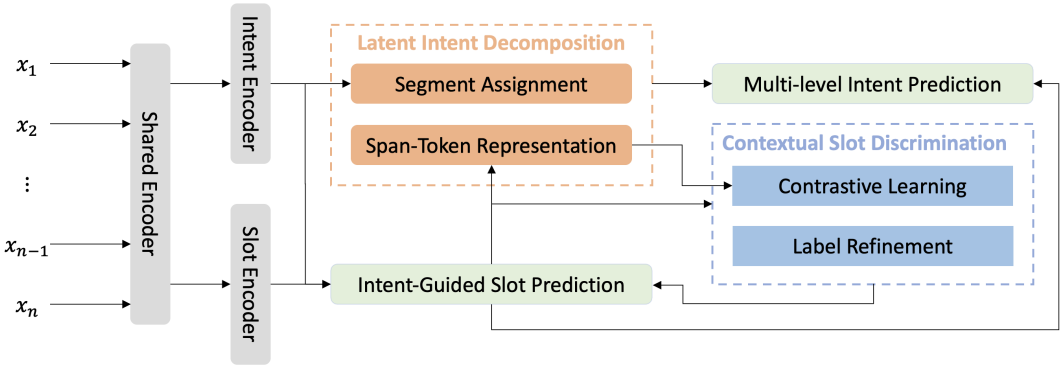

Figure 1: Overview of the proposed LIDS architecture.

demonstrate consistent improvements as compositional complexity increases. Further analysis indicates that distinct intent patterns consistently activate separate latent segments during prediction.

Our main contributions are summarized as follows: (1) We propose LIDS that decomposes encoded features into multiple latent segments and incorporates CSD to stabilize span representations and slot-role discrimination. (2) Experiments and analysis on MixATIS and MixSNIPS show that LIDS improves compositional prediction while learning distinct segment-level activation patterns for different intent combinations.

2 Related Work

2.1 Intent-Slot Interaction and Decomposition

Prior methods mainly focus on structured interaction and decoding strategies. Representative approaches such as AGIF (Qin et al., 2020), GL-GIN (Qin et al., 2021), and Co-guiding Net (Xing and Tsang, 2022) model intent-slot dependency through graph interaction and bidirectional guidance. Later methods extend this direction through multi-view interaction (Tu et al., 2023; Yin et al., 2024), staged structural propagation (Duan et al., 2026), and span-aware decoding (Pham et al., 2023; Xia et al., 2025; Zhang et al., 2025). Generative approaches reformulate SLU as conditional text generation through prompting and instruction-based frameworks (Wu et al., 2022; Xing et al., 2024).

Unlike prior interaction-based approaches that refine interaction over a single global utterance representation, LID decomposes encoded features into multiple latent segments before aggregation. This decomposition strategy is related to soft routing and MoE-style partitioning methods (Shazeer et al., 2017; Fedus et al., 2022), but differs in objective and operating regime. Whereas MoE methods partition computation across sparse expert subnetworks for scalability and conditional computation, LID decomposes encoded features within a fully shared encoder for compositional intent modeling. The latent segments are therefore used to separate overlapping intent patterns during prediction rather than dispatching tokens toward sparse expert computation paths.

2.2 Contrastive Learning

Contrastive learning has recently been introduced into multi-intent SLU to improve semantic consistency and contextual discrimination. BiSLU (Tu et al., 2023) incorporates supervised contrastive objectives and self-distillation into a bidirectional joint model, while subsequent work extends this direction into multi-level and multi-grained formulations (Cheng et al., 2023; Chen et al., 2024). Other methods further explore prototypical objectives, prediction-aware weighting, and label-aware interaction (Song et al., 2023; Zhang et al., 2025), regularizing utterance-level and token-level embedding geometry to encourage aligned representations.

However, semantically similar slot labels with shared boundaries remain difficult to distinguish, especially for hierarchical slot families such as `fromloc.<value>` and `toloc.<value>`. In contrast to prior embedding-level regularization approaches, CSD explicitly targets slot-label confusion through contextual contrastive learning and lightweight refinement over decoded spans, stabilizing decomposition-aware span representations under compositional overlap.

3 Methodology

Figure 1 illustrates the overall architecture of the proposed LIDS. Following prior work (Pham et al.,

2023; Zhang et al., 2025), the model employs a shared encoder together with separate intent and slot encoders. Bidirectional intent-slot interaction is implemented through mutual intent-slot guidance (Xing and Tsang, 2022). Built on this backbone, the model introduces LID, which decomposes encoded features into multiple latent segments to reduce interference under compositional intent prediction. To further enhance span representations and slot discrimination, the model additionally incorporates CSD, which combines contextual contrastive learning (CSD-CL) with lightweight refinement (CSD-Ref).

3.1 Shared and Task-Specific Encoders

Given an utterance $x_i = (w_{i1}, \ldots, w_{iT_i})$, with intent set $y_i \subseteq \mathcal{Y}$ and BIO slot sequence $s_i = (s_{i1}, \ldots, s_{iT_i})$, the shared encoder first produces encoded representations:

$$H_i^{\text{shared}} = \text{Enc}_{\text{shared}}(x_i).$$

The shared representations are then processed by separate intent and slot encoders:

$$H_i^{\text{intent}} = \text{Enc}_{\text{intent}}(H_i^{\text{shared}}),$$

$$H_i^{\text{slot}} = \text{Enc}_{\text{slot}}(H_i^{\text{shared}}).$$

The pooled intent representation is:

$$u_i = \text{Pool}(H_i^{\text{intent}}).$$

3.2 Intent-Guided Slot Prediction

Following (Tu et al., 2023), slot prediction is implemented as a biaffine span classifier conditioned on slot representations and preliminary intent logits:

$$o_i^{\text{soft}} = \psi_{\text{soft}}(u_i),$$

$$o_i^{\text{slot}} = \phi_{\text{slot}}(H_i^{\text{slot}}, o_i^{\text{soft}}), \qquad o_{i,l,r}^{\text{slot}} \in \mathbb{R}^{|\mathcal{S}|}.$$

Let $p_i^{\text{span}}(l, r, s)$ denote the span-level slot distribution obtained from $o_{i,l,r}^{\text{slot}}$ through softmax normalization. Span decoding produces tuples $(l, r, \hat{s}, \kappa)$, corresponding to span boundary, slot label, and decoding confidence. The decoded spans are later used by LID and CSD-Ref, while CSD-CL uses gold non-O spans during training. Slot prediction is optimized using span-level cross entropy:

$$L_{\text{s}} = -\sum_{1 \le l \le r \le T_i} \log p_i^{\text{span}}(l, r, s_{i,l,r}^{\text{span}}).$$

3.3 Latent Intent Decomposition

To mitigate ambiguous intent aggregation from overlapping semantic cues, LID decomposes features into latent segments before aggregation.

3.3.1 Segment Assignment

For each token, a segment scorer produces assignment logits:

$$a_{it} = \psi_{\text{seg}}(h_{it}^{\text{intent}}) \in \mathbb{R}^K,$$

$$\alpha_{itk} = \frac{\exp(a_{itk})}{\sum_{k'=1}^{K} \exp(a_{itk'})},$$

Here, α_{itk} denotes the normalized latent activation of token t for segment k. The assignments are learned jointly from downstream supervision without explicit segment annotations.

3.3.2 Span-Token Representation

LID forms each segment representation from decoded slot spans and residual non-span tokens. Given decoded spans $p = (l, r, \hat{s}, \kappa)$, the span encoder produces:

$$v_i^p = \psi_{\text{span}}\left([h_{il}^{\text{slot}}, h_{ir}^{\text{slot}}, \text{mean}(h_{il:r}^{\text{slot}})]\right),$$

$$\rho_i(p, k) = \frac{1}{r - l + 1} \sum_{t=l}^{r} \alpha_{itk},$$

where $\rho_i(p, k)$ denotes the soft association between span p and latent segment k.

The span representation for segment k is:

$$z_{ik}^{\text{span}} = \frac{\sum_{p \in \mathcal{P}_i} \rho_i(p, k) \kappa_i(p) v_i^p}{\sum_{p \in \mathcal{P}_i} \rho_i(p, k) \kappa_i(p) + \epsilon}.$$

Let $m_{it}^{\text{non}} \in \{0, 1\}$ indicate whether token t is not covered by any decoded span. Residual tokens are pooled using the same latent assignments:

$$z_{ik}^{\text{tok}} = \frac{\sum_{t=1}^{T_i} \alpha_{itk} m_{it}^{\text{non}} h_{it}^{\text{intent}}}{\sum_{t=1}^{T_i} \alpha_{itk} m_{it}^{\text{non}} + \epsilon}.$$

The final segment representation is:

$$r_{ik} = \psi_{\text{proj}}\left([z_{ik}^{\text{span}}, \psi_{\text{tok}}(z_{ik}^{\text{tok}})]\right).$$

During both training and inference, LID operates on decoded spans predicted by the current slot predictor, while decoding confidence κ weights span contributions during aggregation.

3.3.3 Segment Diversity

To encourage specialization across segments, we apply diversity regularization over the assignments:

$$d_{ikt} = \frac{\alpha_{itk}}{\sum_{t'=1}^{T_i} \alpha_{it'k}},$$

$$L_{\text{div}} = \frac{2}{K(K-1)} \sum_{1 \le k < k' \le K} d_{ik}^{\top} d_{ik'}.$$

3.4 Multi-Level Intent Prediction

3.4.1 Competitive Segment Aggregation

Given latent segment representations $\{r_{ik}\}_{k=1}^{K}$, each segment independently predicts intent logits:

$$o_{ik}^{\text{seg}} = \psi_{\text{intent}}(r_{ik}) \in \mathbb{R}^{|\mathcal{Y}|},$$

$$\pi_{iky} = \frac{\exp(o_{iky}^{\text{seg}}/\tau_c)}{\sum_{k'=1}^{K} \exp(o_{ik'y}^{\text{seg}}/\tau_c)}.$$

The resulting local intent logits are:

$$o_{iy}^{\text{local}} = \sum_{k=1}^{K} \pi_{iky} o_{iky}^{\text{seg}}.$$

3.4.2 Slot-Guided Intent Prediction

Following (Xing and Tsang, 2022), global intent logits are computed from the utterance representation and aggregated slot predictions:

$$b_i = \text{Agg}_{\text{slot}}(o_i^{\text{slot}}),$$

$$o_i^{\text{global}} = \psi_{\text{global}}([u_i, \text{softmax}(b_i)]).$$

3.4.3 Intent Fusion

The final prediction combines the local decomposition branch with the slot-guided global branch and is optimized using binary cross entropy:

$$o_i^{\text{final}} = (1-\eta) o_i^{\text{local}} + \eta o_i^{\text{global}},$$

$$L_{\text{i}} = \text{BCE}(o_i^{\text{final}}, y_i).$$

3.5 Contextual Slot Discrimination

To stabilize decomposition-aware span representations and improve slot discrimination under same-boundary slot ambiguity, CSD introduces CSD-CL together with CSD-Ref.

3.5.1 Contextual Slot Representation

Given a decoded span $p = (l, r)$ and context window w, the contextual slot representation is:

$$z_i^p = [v_i^p, c_{i,l}^{\text{left}}, c_{i,r}^{\text{right}}, u_i, \sigma(o_i^{\text{soft}})],$$

$$c_{i,l}^{\text{left}} = \text{mean}(h_{i,l-w:l}^{\text{slot}}),$$

$$c_{i,r}^{\text{right}} = \text{mean}(h_{i,r+1:r+w}^{\text{slot}}).$$

3.5.2 Supervised Contrastive Learning

For each mini-batch, contextual slot representations are constructed from all gold non-O spans. Let $s_p = s_{i,l,r}^{\text{span}}$ denote the gold slot label of span $p = (l, r)$. Positive pairs correspond to spans sharing the same slot label:

$$P_{pq} = \mathbf{1}[s_p = s_q]\mathbf{1}[p \neq q],$$

$$s_{pq} = \frac{\cos(z_i^p, z_i^q)}{\tau_s}.$$

For datasets with hierarchical slot prefixes, slot-family mappings $F(\cdot)$ are constructed from slot-label suffixes, e.g., $F(\texttt{fromloc.<value>}) = F(\texttt{toloc.<value>})$.

Hard-negative pairs correspond to spans sharing the same slot family but different slot labels:

$$H_{pq} = \mathbf{1}[F(s_p) = F(s_q)]\mathbf{1}[s_p \neq s_q],$$

$$w_{pq} = 1 + (\lambda_h - 1) H_{pq}.$$

The supervised contrastive loss is:

$$L_{\text{cl}} = -\frac{1}{|\mathcal{A}|} \sum_{p \in \mathcal{A}} \sum_{q} \tilde{P}_{pq} \log \frac{\exp(s_{pq})}{Z_p},$$

where $\mathcal{A}$ denotes anchors with at least one positive span and

$$Z_p = \sum_{q' \neq p} w_{pq'} \exp(s_{pq'}).$$

3.5.3 Label Refinement

LIDS further applies a lightweight refinement module after span decoding:

$$o_i^p = \psi_{\text{refine}}(z_i^p), \qquad o_i^p \in \mathbb{R}^{|\mathcal{S}|}.$$

The refinement branch is optimized using cross entropy over gold slot labels:

$$L_{\text{ref}} = -\sum_{p} \log \text{softmax}(o_i^p)_{s_p}.$$

At decoding time, refinement produces a residual correction over decoded spans:

$$o_{i,l,r}^{\text{slot,final}} = o_{i,l,r}^{\text{slot}} + \alpha_{\text{refine}} \psi_{\text{refine}}(z_i^p).$$

3.6 Training Objective

The overall objective combines slot prediction, intent prediction, latent segment regularization, contrastive learning, and label refinement:

$$L = L_{\text{s}} + L_{\text{i}} + \lambda_{\text{div}} L_{\text{div}} + \lambda_{\text{cl}} L_{\text{cl}} + \lambda_{\text{ref}} L_{\text{ref}}.$$

Table 1: Statistics of MixATIS and MixSNIPS datasets.

Dataset	Train	Dev	Test	Slots	Intents
MixATIS	13,162	759	828	78	18
MixSNIPS	39,776	2,198	2,199	39	7

4 Experimental Setup

4.1 Datasets

We evaluate LIDS on the standardized benchmarks MixATIS and MixSNIPS (Qin et al., 2020), which are widely adopted in prior multi-intent SLU studies and enable consistent comparison across different modeling paradigms (Wu et al., 2025). MixATIS focuses on airline travel queries with substantial intent overlap and hierarchical slot ambiguity, making it particularly challenging for intent prediction and slot discrimination. In contrast, MixSNIPS contains more diverse domains with comparatively lower intent overlap and slot ambiguity. We use the standard train, development, and test splits as summarized in Table 1.

4.2 Implementation Details

For fair comparison across methods, all experiments use pretrained RoBERTa-base (Liu et al., 2019) as the shared encoder. $\text{Enc}_{\text{intent}}$ and Enc_{slot} use Transformer encoders. The slot prediction module ϕ_{slot} uses a biaffine span classifier, while all $\psi(\cdot)$ modules denote MLP-based prediction heads. The model is optimized using AdamW with learning rate 5×10^{-5}, batch size 64, and a maximum of 30 training epochs. Early stopping is applied on the development set, and hyperparameters are tuned using random search. Default and tuning hyperparameters are provided in Appendix A.1.

Following (Cheng et al., 2023; Chen et al., 2024), data augmentation is additionally applied to MixATIS only due to its smaller training size and larger slot and intent label spaces compared with MixSNIPS. Hard-negative weighting is likewise applied only on MixATIS, where hierarchical slot families are explicitly defined, while MixSNIPS uses standard supervised contrastive learning without family-level hard negatives. Additional augmentation details and slot-family mappings are provided in Appendix A.3.

4.3 Baselines

We compare LIDS against representative baselines spanning graph reasoning, intent-slot interaction, decoding-oriented methods, and contrastive learning approaches. Non-pretrained baselines include AGIF (Qin et al., 2020), GL-GIN (Qin et al., 2021), and Co-guiding Net (Xing and Tsang, 2022). Pre-trained baselines include MISCA (Pham et al., 2023), BiSLU (Tu et al., 2023), Uni-MIS (Yin et al., 2024), PACL (Chen et al., 2024), SpanDec (Xia et al., 2025), and LCAN (Zhang et al., 2025), which incorporate co-attention, span prediction, multi-view reasoning, contrastive learning, and label-aware interaction. Recent multi-intent SLU studies have also explored instruction-based and generative formulations (Wu et al., 2022; Xing et al., 2024). However, classification-based models remain more computationally efficient while preserving explicit structured prediction and token-level alignment. Since this work focuses on decomposition-aware representation modeling within encoder-based joint SLU, we therefore restrict comparisons to encoder-based baselines.

For all baselines, we report the published results from the original papers on the standard MixATIS and MixSNIPS splits.

Table 2: Main results on MixATIS and MixSNIPS. Bold indicates the best result and underline indicates the second-best result within each group.

Models	MixATIS			MixSNIPS		
	Slot (F1)	Intent (Acc)	Sem (Acc)	Slot (F1)	Intent (Acc)	Sem (Acc)
w/o Pre-trained Language Models						
AGIF	86.7	74.4	40.8	94.2	95.1	74.2
GL-GIN	88.3	76.3	43.5	94.9	95.6	75.4
Co-guiding	89.8	79.1	51.3	95.1	97.7	77.5
w/ Pre-trained Language Models						
MISCA	90.0	80.6	58.1	96.1	97.6	83.1
BiSLU	89.4	81.5	51.5	**97.2**	97.8	**85.4**
Uni-MIS	88.3	78.5	52.5	96.4	97.2	83.4
PACL	87.3	<u>81.9</u>	50.4	96.8	96.9	85.1
SpanDec	89.7	80.6	53.4	<u>97.1</u>	<u>98.0</u>	84.7
LCAN	**92.4**	81.3	<u>59.5</u>	96.9	**98.7**	<u>85.3</u>
LIDS (Ours)	<u>91.4</u>	**83.5**	**60.1**	96.5	96.9	84.1

5 Results and Analysis

5.1 Main Results

Table 2 presents the main results on MixATIS and MixSNIPS. On MixATIS, LIDS achieves the highest intent accuracy of 83.5% and the best semantic accuracy of 60.1%, outperforming LCAN (59.5%). Although LCAN obtains the highest slot F1, LIDS achieves the second-highest slot F1 (91.4%) together with the strongest intent and semantic performance overall. These results suggest that improving intent disentanglement contributes substantially

Table 3: Compositional complexity and slot ambiguity analysis on MixATIS. Lower error values are better.

Model	Semantic Acc.			Same-boundary Error	
	$k=1$	$k=2$	$k=3$	Overall	Family
Backbone	75.5	45.2	27.0	0.115	0.072
LIDS	**82.5**	**61.6**	**40.5**	**0.068**	**0.052**

to compositional understanding beyond optimizing local slot prediction alone.

On MixSNIPS, it achieves competitive performance with 84.1% semantic accuracy. Compared with MixATIS, the relative benefit of LID becomes less pronounced, which is consistent with dataset characteristics. MixATIS contains stronger intent overlap together with larger slot and intent label spaces, increasing semantic interference and slot ambiguity, whereas MixSNIPS contains more training samples, fewer labels, and more heterogeneous domains.

Overall, the results indicate that LIDS is particularly effective under challenging compositional settings with overlapping intent structures, supporting the hypothesis that latent decomposition helps reduce interference inside shared utterance representations.

5.2 Compositional Complexity and Slot Ambiguity Analysis

Table 3 presents compositional complexity and slot ambiguity analysis on MixATIS. Here, $k = 1$, $k = 2$, and $k = 3$ denote utterances containing one, two, and three intents, respectively. As compositional complexity increases, the gains from LIDS become substantially larger, indicating that the proposed model is particularly effective under overlapping multi-intent settings rather than simpler single-intent compositions. In particular, semantic accuracy improves from 27.0% to 40.5% on three-intent utterances, corresponding to a 50.0% relative improvement over the backbone. In contrast, the improvement on single-intent utterances remains comparatively smaller, suggesting that the primary benefit of LIDS emerges under stronger compositional interference.

The model also reduces same-boundary slot confusion under compositional overlap. Compared with the backbone, overall same-boundary error decreases from 0.115 to 0.068, corresponding to a 41.1% relative reduction, while family-level ambiguity decreases from 0.072 to 0.052. These results suggest that LIDS produces more discriminative representations for semantically overlapping slot candidates sharing similar contextual boundaries.

Table 4: Sensitivity analysis of LIDS under different hyperparameter settings on MixATIS. The shaded row denotes the default configuration used in the final model.

K	τ_c	η	τ_s	**Slot**	**Intent**	**Sem**
3	0.5	0.35	0.05	**91.4**	83.5	**60.1**
4	0.5	0.35	0.05	91.3	**83.7**	60.0
3	0.3	0.35	0.05	91.1	83.2	59.1
3	0.7	0.35	0.05	91.3	83.5	59.9
3	0.5	0.5	0.05	91.2	83.3	59.4

5.3 Sensitivity Analysis

Table 4 presents the sensitivity analysis of LIDS under different latent decomposition and aggregation configurations on MixATIS. We vary the latent segment count K, the competitive aggregation temperature τ_c, and the local-global fusion weight η, while keeping the remaining hyperparameters fixed to the default configuration. To analyze local robustness around the selected operating point, sensitivity experiments initialize from the default trained configuration while varying only the target hyperparameters.

The results show that LIDS remains stable once sufficient decomposition capacity is available. Performance stays highly consistent between $K = 3$ and $K = 4$, suggesting that the model does not rely on highly sensitive segment configurations. Increasing K beyond the typical number of intents does not substantially degrade performance, indicating that additional latent segments can absorb residual contextual information rather than forcing unstable intent partitioning. In contrast, preliminary experiments with smaller segment counts (e.g., $K = 2$) produced noticeably weaker compositional separation, indicating that insufficient latent capacity limits segment-to-intent specialization under overlapping intent structures.

The remaining hyperparameters produce only minor performance variations around the default configuration. Varying τ_c and η changes semantic accuracy only marginally, suggesting that the decomposition and aggregation mechanisms remain stable under moderate perturbations.

5.4 Decomposition Stability Analysis

LID is intentionally trained using predicted spans rather than gold spans, since only predicted spans

are available during inference. This avoids train-test mismatch between training and deployment during latent decomposition and aggregation.

We further investigated whether alternative span-training schedules could stabilize standalone LID without CSD. Specifically, we experimented with (i) predicted-span-only training, (ii) gold-span warm-up during early epochs followed by predicted spans, and (iii) gradual gold/predicted span mixing before switching to fully predicted spans. However, all standalone LID variants remained unstable and failed to produce substantial semantic improvements over the backbone.

These observations suggest that the main challenge is not solely span-boundary mismatch, but the lack of stable span-level semantic discrimination during decomposition and aggregation. CSD therefore plays a structural role in the model by stabilizing the span representations consumed by LID, rather than acting as a purely auxiliary refinement component. The results further indicate that LID and CSD operate as coupled mechanisms rather than independently additive modules.

5.5 Latent Segment Analysis

A central motivation of LID is to reduce interference inside shared utterance representations. However, latent decomposition may suffer from trivial collapse, where most tokens concentrate into a single dominant segment, or fail to separate different intent structures. To evaluate the learned latent decomposition, we analyze token-to-segment assignments, intent-level segment specialization, and assignment confidence over both span and token representations.

Given token-level segment assignments α_{itk}, we evaluate whether latent segments remain diverse and aligned with different intent structures using several complementary metrics.

Assignment Entropy. Normalized entropy over token-to-segment assignments is defined as:

$$H_i = -\frac{1}{T_i \log K} \sum_{t=1}^{T_i} \sum_{k=1}^{K} \alpha_{itk} \log \alpha_{itk}.$$

Effective Segment Usage. The effective number of active latent segments follows entropy-based usage estimation (Hill, 1973):

$$K_{\text{eff}} = \exp\left(-\sum_{k=1}^{K} \bar{\alpha}_{ik} \log \bar{\alpha}_{ik} \right),$$

Table 5: Latent segment specialization, decomposition, and token assignment analysis for LIDS.

Dataset	Entropy	K_{eff}	Peak	R_{diff}	Span	Token
MixATIS	0.088	2.119	0.723	0.853	0.949	0.964
MixSNIPS	0.001	2.920	0.417	0.910	0.999	0.999

$$\bar{\alpha}_{ik} = \frac{1}{T_i} \sum_{t=1}^{T_i} \alpha_{itk}.$$

Peak Segment Mass. Maximum average token mass assigned to any latent segment is defined as:

$$M_i = \max_k \bar{\alpha}_{ik}.$$

Different Segment Rate. To analyze whether different intents are assigned to different dominant latent segments, let

$$k^*(y) = \arg\max_k \pi_{iky}$$

denote the dominant segment selected by competitive aggregation. The different-segment rate is:

$$R_{\text{diff}} = \frac{\sum_{(y_a, y_b)} \mathbb{I}[k^*(y_a) \neq k^*(y_b)]}{|\mathcal{P}|},$$

where $\mathcal{P}$ denotes all unordered intent pairs within the same utterance.

Span and Token Assignment Confidence. To analyze whether LID relies only on decoded spans or also utilizes broader contextual information, dominant assignment confidence is separately defined over span-covered and residual token representations:

$$\text{SpanMass}_i = \frac{\sum_{t=1}^{T_i} (1 - m_{it}^{\text{non}}) \max_k \alpha_{itk}}{\sum_{t=1}^{T_i} (1 - m_{it}^{\text{non}}) + \epsilon},$$

$$\text{TokenMass}_i = \frac{\sum_{t=1}^{T_i} m_{it}^{\text{non}} \max_k \alpha_{itk}}{\sum_{t=1}^{T_i} m_{it}^{\text{non}} + \epsilon},$$

where $m_{it}^{\text{non}} = 1$ if token t is not covered by any decoded span.

Table 5 summarizes the latent segment analysis results. Across both datasets, the effective number of active segments remains substantially larger than 1, with K_{eff} ranging from approximately 2.1 to 2.9, indicating that latent decomposition avoids trivial single-segment collapse and actively utilizes multiple latent segments. Entropy values remain

relatively low, while peak segment mass stays below complete concentration into a single latent segment, suggesting stable and non-collapsed token assignments across segments. Most importantly, the different segment rate remains consistently high across both datasets, reaching 0.853 on MixATIS and 0.910 on MixSNIPS, indicating that different intents within the same utterance are typically assigned to different dominant latent segments during competitive aggregation. Token assignment confidence is also comparable to span assignment confidence, suggesting that LID utilizes not only decoded spans but also broader contextual information. Overall, the results support the hypothesis that latent decomposition helps reduce interference between overlapping intent structures before aggregation.

Figure 2 visualizes token-to-segment assignments for representative utterances. Distinct intent regions are consistently assigned to different latent segments rather than collapsing into a single shared representation. The examples further show that $K = 3$ provides sufficient latent capacity to separate two-intent compositions while still preserving additional segments for residual contextual information rather than forcing strict one-segment-per-intent assignments. Latent decomposition is also not driven solely by localized spans. Residual tokens such as *from*, *to*, *ticket*, and *flight* receive strong segment assignments, consistent with the token assignment analysis in Table 5. Taken together, the visualizations suggest that LID learns specialized and non-collapsed latent segments aligned with multi-intent structure.

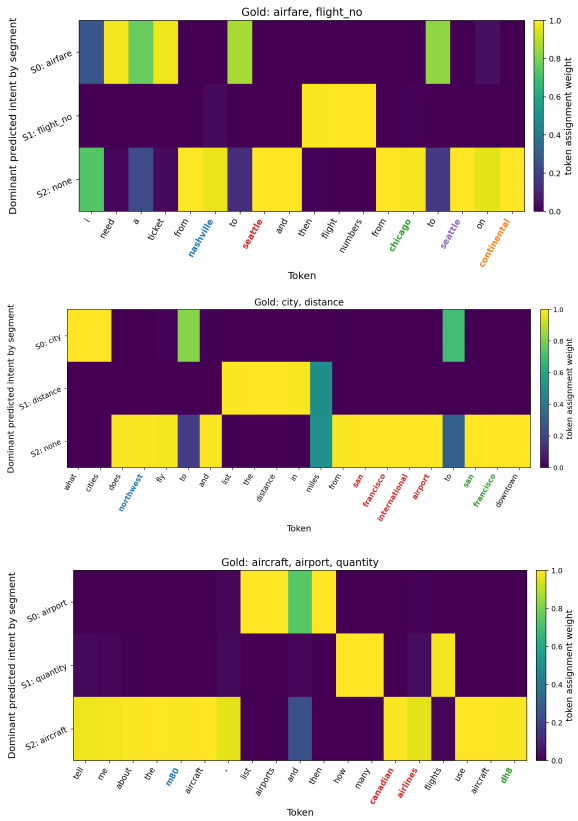

Figure 2: Representative token-to-segment assignment heatmaps produced by LIDS on MixATIS. Different intent regions are assigned to different dominant latent segments, while residual tokens additionally contribute beyond localized slot spans.

6 Conclusion

We presented LIDS for multi-intent SLU centered on latent intent decomposition. Unlike prior approaches that refine interaction, graph propagation, or decoding over a single global utterance representation, LID decomposes contextual representations into latent segments before intent aggregation, mitigating representational interference under compositional intent structures. LIDS further incorporates CSD, combining contextual contrastive learning with lightweight refinement over decoded spans to stabilize span representations and improve fine-grained slot discrimination. Experimental results on MixATIS and MixSNIPS demonstrate consistent gains in compositional understanding, with the largest improvements observed on MixATIS under severe intent overlap and slot ambiguity. Additional analysis shows that LID learns specialized and non-collapsed latent segments aligned with different intent structures.

Overall, the results highlight the importance of reducing representational interference for robust compositional understanding. A promising future direction is to extend LID toward adaptive segment structures and sparse MoE routing, where latent segment assignments provide task-aware signals for expert selection. Such integration may improve scalability for large-label or multi-domain SLU settings while preserving decomposition-aware prediction under high compositional complexity.

7 Limitations

LIDS is evaluated only on MixATIS and MixSNIPS, and additional validation on larger real-world SLU benchmarks would further strengthen the generality of the proposed decomposition model. The improvements from LIDS are also more pronounced on datasets with substantial in-

tent overlap and slot ambiguity, while gains on simpler settings such as MixSNIPS remain comparatively smaller. In addition, family-level hard-negative weighting is currently applicable only to datasets with hierarchical slot-prefix structures such as MixATIS, while semantic augmentation was empirically beneficial only on MixATIS in our experiments.

In addition, the current formulation uses a fixed latent segment count K, which may not optimally adapt to varying compositional complexity across utterances. The decomposition quality additionally depends on the quality of decoded slot spans, since LID consumes slot predictions during aggregation.

Finally, this work focuses primarily on compositional representation learning rather than scalability or inference efficiency. Future work may explore adaptive segment structures, sparse MoE-based routing, and robustness under ASR noise.

Acknowledgments

References

Guanhua Chen, Yutong Yao, Derek F Wong, and Lidia S Chao. 2024. A two-stage prediction-aware contrastive learning framework for multi-intent nlu. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1778–1788.

Xuxin Cheng, Wanshi Xu, Zhihong Zhu, Hongxiang Li, and Yuexian Zou. 2023. Towards spoken language understanding via multi-level multi-grained contrastive learning. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 326–336.

Shijie Duan, Long Yu, and Shengwei Tian. 2026. A two-stage line graph reasoning framework for multi-intent spoken language understanding. *Neurocomputing*, page 133053.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*.

Mark O Hill. 1973. Diversity and evenness: a unifying notation and its consequences. *Ecology*, 54(2):427–432.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

Thinh Pham, Chi Tran, and Dat Quoc Nguyen. 2023. Misca: A joint model for multiple intent detection and slot filling with intent-slot co-attention. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12641–12650.

Libo Qin, Fuxuan Wei, Tianbao Xie, Xiao Xu, Wanxiang Che, and Ting Liu. 2021. Gl-gin: Fast and accurate non-autoregressive model for joint multiple intent detection and slot filling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 178–188.

Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu. 2020. Agif: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1807–1816.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*.

Yurun Song, Junchen Zhao, Spencer Koehler, Amir Abdullah, and Ian Harris. 2023. Pcmid: Multi-intent detection through supervised prototypical contrastive learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9481–9495.

Nguyen Anh Tu, Hoang Thi Thu Uyen, Tu Minh Phuong, and Ngo Xuan Bach. 2023. Joint multiple intent detection and slot filling with supervised contrastive learning and self-distillation. *Preprint*, arXiv:2308.14654.

Di Wu, Ruiyu Fang, Liting Jiang, Shuangyong Song, Xiaomeng Huang, Shiquan Wang, Zhongqiu Li, Lingling Shi, Mengjiao Bao, Yongxiang Li, and 1 others. 2025. Multi-intent spoken language understanding: a survey of methods, trends, and challenges. *Vicinagearth*, 2(1):20.

Yangjun Wu, Han Wang, Dongxiang Zhang, Gang Chen, and Hao Zhang. 2022. Incorporating instructional prompts into a unified generative framework for joint multiple intent detection and slot filling. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 7203–7208.

Ying Xia, Zhen Xiong, Kefan Shen, Zhihong Zhu, Shaorong Xie, and Wei Liu. 2025. Rethinking decoding in multi-intent spoken language understanding. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Bowen Xing, Lizi Liao, Minlie Huang, and Ivor Tsang. 2024. Dc-instruct: An effective framework for generative multi-intent spoken language understanding. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 14520–14534.

Table 6: Default architectural parameterization of LIDS. $|\mathcal{Y}|$, $|\mathcal{S}|$, and d denote the numbers of intent labels, slot labels, and the final-layer hidden dimension of RoBERTa, respectively.

Module	Implementation
ψ_{soft}	`Linear(d, |Y|)`
ϕ_{slot}	biaffine span classifier with `Linear(d, |Y|)`
ψ_{span}	`Linear(3d, d)`
ψ_{proj}	`Linear(2d, d)`
ψ_{tok}	`Linear(d, d)`
ψ_{intent}	`Linear(d, |Y|)`
Agg_{slot}	Non-param confidence-weighted aggregation
ψ_{global}	`Linear(d + |S|, |Y|)`
ψ_{refine}	`Linear(3d, d)`$\rightarrow$ `Linear(d, |S|)`

Bowen Xing and Ivor Tsang. 2022. Co-guiding net: Achieving mutual guidances between multiple intent detection and slot filling via heterogeneous semantics-label graphs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 159–169.

Shangjian Yin, Peijie Huang, and Yuhong Xu. 2024. Uni-mis: United multiple intent spoken language understanding via multi-view intent-slot interaction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 19395–19403.

Shuli Zhang, Zhiqiang You, Xiaoxiang Qi, Peng Liu, Gaode Wu, Kan Xia, and Shenguang Huang. 2025. Lcan: A label-aware contrastive attention network for multi-intent recognition and slot filling in task-oriented dialogue systems. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 25603–25612.

A Appendix

A.1 Additional Implementation and Training Details

Predicted vs. Gold Span Training. LIDS is trained using predicted spans rather than gold spans, since only predicted spans are available during inference. This avoids train-test mismatch during latent decomposition and aggregation. In contrast, CSD-CL constructs contrastive supervision from gold non-`O` spans during training because its objective focuses on span-level representation regularization rather than inference-time decomposition.

Default Parameters. Table 6 lists the default architectural parameterization of LIDS. All encoder hidden dimensions follow RoBERTa-base with hidden size $d = 768$. The preliminary intent classifier ψ_{soft} and segment-level intent classifier ψ_{intent} both use lightweight linear prediction heads, where the latter shares parameters with ψ_{soft}.

The slot predictor ϕ_{slot} is implemented as an intent-guided biaffine span classifier conditioned on preliminary intent logits. Span representations are encoded through ψ_{span}, while latent segment fusion combines span-level and residual token-level evidence through ψ_{proj} and ψ_{tok}. Global intent prediction ψ_{global} aggregates utterance-level intent representations together with confidence-weighted slot evidence produced by the non-parametric aggregation module Agg_{slot}.

For contextual slot discrimination, the refinement module ψ_{refine} uses a lightweight two-layer MLP operating on span, utterance, and intent-context representations to produce residual slot-label corrections over decoded spans.

Tuning Parameterization. Table 7 presents the hyperparameter search space together with the final selected configurations for MixATIS and MixSNIPS. Backbone tuning controls the depth and capacity of the task-specific Transformer encoders through Enc_{layer}, Enc_{head}, and Enc_{ffn}.

The LID stage is separated into architectural and control components. Architectural tuning optimizes the depth and width of the segment decomposition module together with the latent segment count K, while control tuning adjusts the competitive aggregation temperature τ_c and local-global fusion weight η.

After selecting the LID configuration, CSD tunes the contrastive temperature τ_s. The final stage then optimizes the auxiliary objective weights λ_{div}, λ_{cl}, and λ_{ref} for joint training.

Computational Cost. All experiments are conducted on a single NVIDIA RTX 5090 GPU with 32GB memory. LIDS contains approximately 275M parameters on MixATIS and 281M parameters on MixSNIPS. Training costs about 1.0 second per optimization step, corresponding to approximately 331 seconds per MixATIS epoch and 624 seconds per MixSNIPS epoch. At inference time, evaluation takes about 0.014 seconds per sample.

Although the biaffine slot predictor enumerates a span grid, the practical refinement overhead remains small. The average number of valid span candidates is 298.8 on MixATIS and 283.4 on MixSNIPS, but only decoded non-`O` spans are passed to CSD-Ref. This results in only 4.55 refinement candidates per utterance on MixATIS and 5.20 on MixSNIPS on average. Most additional computation therefore comes from shared/task-specific encoding and span scoring, while refinement oper-

Table 7: Hyperparameter search space and selected configurations for LIDS on MixATIS and MixSNIPS.

Components	Module	Tuning Parameters	MixATIS	MixSNIPS
Backbone	Enc_{layer}	$\{2, 3, 4\}$	2	3
	Enc_{head}	$\{4, 8\}$	4	4
	Enc_{ffn}	$\{2\times, 4\times\}$	$2\times$	$2\times$
LID	$\psi_{\text{seg,layer}}$	$\{2, 3, 4\}$	2	2
	$\psi_{\text{seg,width}}$	$\{512, 768\}$	512	512
	K	$\{2, 3, 4\}$	3	3
	τ_c	$\{0.3, 0.5, 0.6, 0.7\}$	0.5	0.5
	η	$\{0.35, 0.5, 0.65, 0.8\}$	0.35	0.35
CSD	τ_s	$\{0.05, 0.07, 0.1, 0.15\}$	0.05	0.05
Loss	λ_{div}	$\{0.01, 0.03, 0.05\}$	0.03	0.03
	λ_{cl}	$\{0.1, 0.15, 0.2\}$	0.15	0.15
	λ_{ref}	$\{0.1, 0.15, 0.2\}$	0.15	0.15

Table 8: Component interaction and decomposition stability analysis on MixATIS.

Model	Slot	Intent	Sem
Backbone	85.8	78.7	46.0
+ LID (Pred)	85.8	80.0	47.0
+ LID (Gold)	86.4	77.9	47.6
+ LID (Mix)	87.0	78.6	49.4
+ CSD-CL	89.8	82.9	55.4
+ CSD-Ref	86.6	80.9	49.4
LIDS	**91.4**	**83.5**	**60.1**

ates only on a small decoded subset rather than the full span grid.

A.2 Component Interaction and Decomposition Stability

Unlike conventional additive ablations, LID and CSD operate as coupled mechanisms under predicted-span training. Table 8 therefore analyzes decomposition stability and component interaction behavior rather than independent additive contributions.

We first analyze whether standalone LID can improve compositional separation under different span-training schedules. *Pred* denotes predicted-span-only training, *Gold* uses gold spans during early training before switching to predicted spans, and *Mix* gradually mixes gold and predicted spans before fully predicted-span training. Although the different schedules produce minor variations, all standalone LID variants remain relatively unstable and achieve only limited semantic improvements over the backbone.

The results suggest that the main challenge is not solely train-test mismatch between gold and predicted spans, but insufficient span-level semantic discrimination during decomposition and aggregation. In particular, predicted-span decomposition introduces noisy and semantically ambiguous span representations, which can destabilize latent segment specialization under compositional overlap.

Adding CSD-CL on top of LID substantially improves all evaluation metrics, especially semantic accuracy, indicating that contextual contrastive learning plays an important role in stabilizing decomposition-aware span representations. In contrast, adding CSD-Ref alone produces comparatively smaller gains, suggesting that representation stabilization is more critical than lightweight post-refinement alone. The full LIDS model achieves the best overall performance, supporting the hypothesis that LID and CSD function as coupled mechanisms rather than independently additive modules.

A.3 Data Augmentation and Slot-Family Construction

Following prior multi-intent SLU studies (Cheng et al., 2023; Chen et al., 2024), we apply semantic data augmentation to improve compositional diversity and slot robustness under limited supervision.

Compositional augmentation. Additional multi-intent utterances are generated by concatenating two or three training utterances using simple connector tokens such as and. The augmented intent set is formed through intent union, while slot annotations are transferred to the new token positions and connector tokens receive the O label. To further increase lexical diversity, a subset of generated examples additionally applies same-label slot-value replacement using alternative spans sampled from training examples with identical slot labels. Duplicate generations together with excessive length

Table 9: Slot-family mappings used for hard-negative weighting on MixATIS.

Family	Slot Labels
airport_code	airport_code, fromloc.airport_code, stoploc.airport_code, toloc.airport_code
airport_name	airport_name, fromloc.airport_name, stoploc.airport_name, toloc.airport_name
city_name	city_name, fromloc.city_name, stoploc.city_name, toloc.city_name
date_relative	arrive_date.date_relative, depart_date.date_relative, return_date.date_relative
day_name	arrive_date.day_name, day_name, depart_date.day_name
day_number	arrive_date.day_number, depart_date.day_number, return_date.day_number, day_number
end_time	arrive_time.end_time, depart_time.end_time
month_name	arrive_date.month_name, depart_date.month_name, month_name, return_date.month_name
period_mod	arrive_time.period_mod, depart_time.period_mod
period_of_day	arrive_time.period_of_day, depart_time.period_of_day, period_of_day
start_time	arrive_time.start_time, depart_time.start_time
state_code	fromloc.state_code, state_code, stoploc.state_code, toloc.state_code
state_name	fromloc.state_name, state_name, toloc.state_name
time	arrive_time.time, depart_time.time, time
time_relative	arrive_time.time_relative, depart_time.time_relative, time_relative
today_relative	arrive_date.today_relative, depart_date.today_relative, today_relative

and slot-density deviations are filtered to preserve semantic consistency.

Slot-family augmentation. Semantically related slot groups with limited lexical diversity are further augmented through compatible slot-value replacement within the same semantic family while preserving the surrounding utterance structure.

During training, mini-batches dynamically combine original and augmented examples, allowing the model to observe different compositional variations across epochs while maintaining exposure to the original training distribution.

Slot-Family Construction. For MixATIS, slot families are constructed automatically from slot-label suffixes. Slot labels sharing the same semantic suffix are grouped into the same family for hard-negative weighting in CSD-CL. For example, directional slot variants such as `fromloc.city_name` and `toloc.city_name` belong to the shared `city_name` family. The complete mapping is shown in Table 9.

MixSNIPS does not contain consistent hierarchical slot-prefix structures and therefore does not use family-level hard-negative weighting.

A.4 Additional Decomposition Visualizations

Figure 3 further analyzes the effect of latent segment capacity by visualizing the same utterance structures under different K configurations, including both smaller and larger capacities compared with the representative $K = 3$ examples in Figure 2.

When the capacity is insufficient ($K = 2$) under highly overlapping three-intent compositions, different intent regions become less distinctly separated within the decomposition. However, $K = 2$ can still maintain reasonable separation for other three-intent utterances with weaker overlap, suggesting that the degradation mainly emerges under stronger compositional interference.

Increasing the capacity to $K = 4$ preserves stable latent decomposition even when the number of segments exceeds the number of active intents. Additional segments are typically assigned to residual contextual tokens or supporting semantic regions rather than destabilizing the dominant intent decomposition. This observation is consistent with the sensitivity analysis, where performance remains stable once sufficient decomposition capacity becomes available.

Overall, the additional visualizations further support the hypothesis that latent decomposition capacity plays an important role in compositional separation, while excess capacity can be utilized for contextual absorption without forcing unstable intent partitioning.

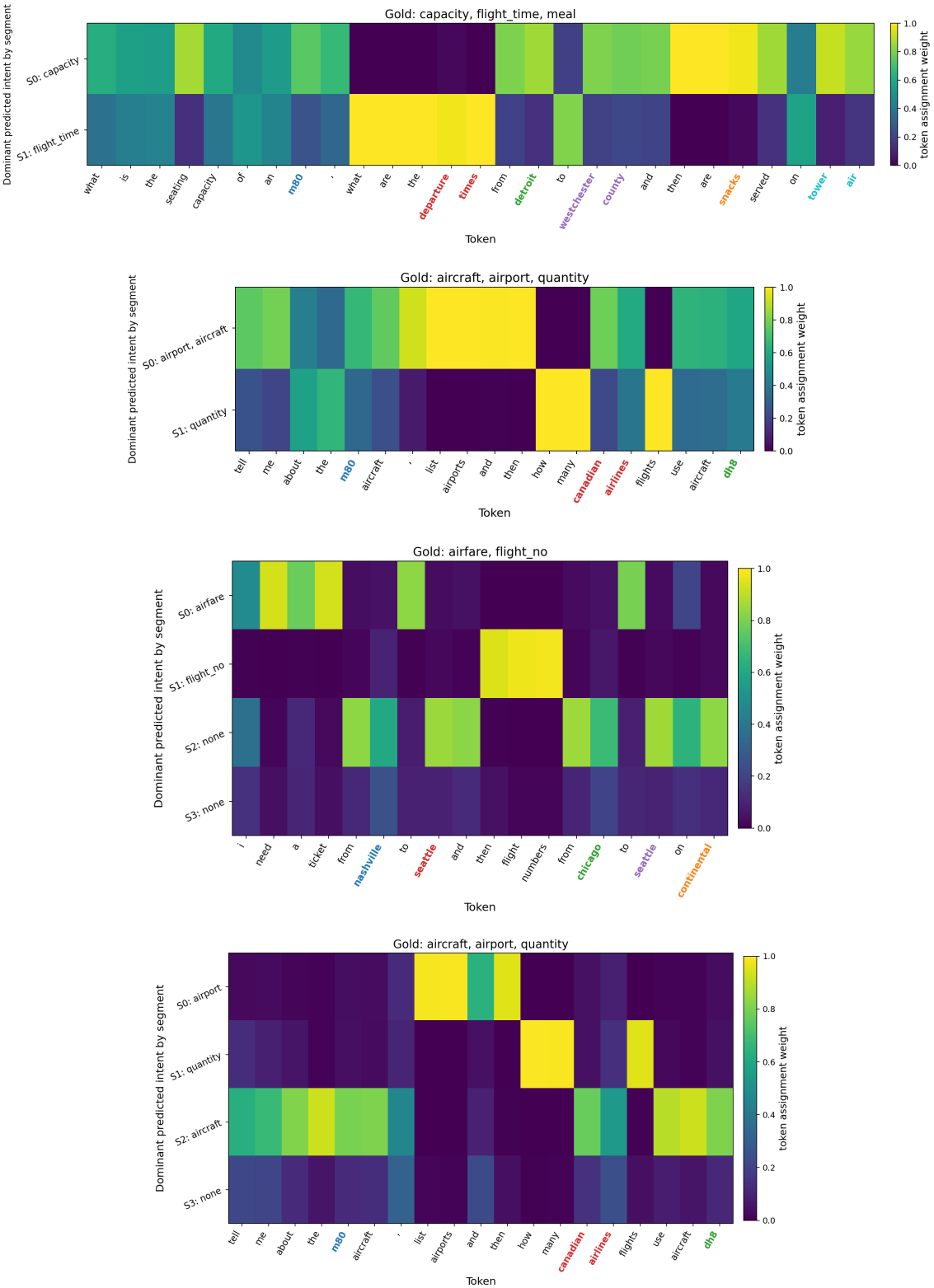

Figure 3: Additional decomposition visualizations under different latent segment capacities. Top and middle: $K = 2$. Bottom: $K = 4$.