scientific reports

Check for updates

OPEN

Siamese-based metric joint learning for intent detection and slot filling using triplet loss optimization

Yusuf Idris Muhammad[1,2]✉, Naomie Salim[1], Anazida Zainal[1], Maged Nasser[3], Ahmad Sobri Hashim[3], Sharin Hazlin Huspi[1] & Yunusa Adamu Bena[4]

Spoken language understanding (SLU) relies on intent detection and slot filling to interpret user utterances accurately. However, existing joint learning frameworks struggle to generalize across minority intent classes and paraphrase queries. They depend heavily on token-level embeddings and classification losses such as cross-entropy, which do not explicitly model semantic similarity. To address this limitation, this study proposes a Siamese-Based Metric Joint Learning model for Intent Detection and Slot Filling (SBJLIS). The model uses triplet loss optimization to enhance semantic distance learning between utterances. Unlike standard cross-entropy training, triplet loss enforces separation between dissimilar classes and brings semantically related sentences closer in the embedding space. This approach improves both discrimination and generalization. SBJLIS employs a unified two-stage SLU framework. The first stage uses a Siamese network for metric-based similarity learning. The second stage integrates an attention-based joint decoder for simultaneous intent detection and slot filling. By aligning embedding geometry with multi-task objectives, the model improves semantic discrimination and robustness to class imbalance and linguistic variation. Experimental results show that SBJLIS achieves 98.87% accuracy and 98.60% F1-score on the ATIS dataset, and 99.61% accuracy and 98.68% F1-score on SNIPS, outperforming all existing baselines. These findings confirm that metric-based similarity learning offers an interpretable and generalizable foundation for advanced conversational AI systems.

Keywords Intent detection, Slot filling, Joint learning, Siamese, Triplet loss

Spoken language understanding (SLU) is a core component of modern conversational AI, enabling systems such as virtual assistants and chatbots to interpret user utterances and extract structured meaning from natural language input[1,2]. It comprises two interdependent tasks: intent detection, which identifies the user's goal, and slot filling, which extracts semantic entities related to that intent[3–5]. Early SLU systems treated these tasks independently in a pipeline configuration, where errors in intent detection propagated to slot filling, reducing overall performance[6]. To overcome this limitation, joint learning frameworks were introduced to model the interdependence between the two tasks, improving contextual representation and performance. Architectures such as the slot-gated[7], bidirectional joint networks[8], and co-interactive models like CEA-Net[9] demonstrated improved performance by sharing contextual representations.

Despite these advances, joint learning models continue to face challenges. Many struggle to generalize to minority intent classes, where frequent classes dominate[6]. Furthermore, models often fail to handle semantic variation and paraphrasing effectively; for example, the utterances "Book a flight to Paris" and "Reserve a ticket for Paris" may express the same intent but yield inconsistent predictions. Most existing frameworks rely on token-level embeddings and cross-entropy-based optimization, which overfit to frequent patterns and fail to capture broader semantic similarity across utterances. While metric learning methods such as Siamese networks with triplet loss have achieved strong results in domains like face recognition[10] and speaker verification[11], their integration into end-to-end joint SLU architectures remains limited and underexplored.

[1]Faculty of Computing, Universiti Teknologi, 81310 Skudai, Johor, Malaysia. [2]Department of Computer Science, Sa'adatu Rimi College of Education, Kumbotso 3218, Kano, Nigeria. [3]Department of Computing, Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Malaysia. [4]Department of ICT, Faculty of Engineering, Kebbi State University of Science and Technology, Aliero, Nigeria. ✉email: muhammadidris@graduate.utm.my

To address these limitations, this study proposes the Siamese-Based Metric Joint Learning for Intent Detection and Slot Filling (SBJLIS). SBJLIS integrates a Siamese network encoder for metric learning with an attention-based joint decoder, enforcing similarity-based distance constraints to shape a semantically structured embedding space. This design explicitly aligns embedding geometry with multi-task objectives, allowing the model to discriminate between semantically close and distant intents while improving robustness to class imbalance and linguistic variation. Unlike prior contrastive-learning or attention-only approaches, SBJLIS establishes a unified metric–attention model that couples Siamese-based representation learning with joint intent–slot decoding.

The main contributions of this study are as follows:

- A unified Siamese-based joint learning framework that integrates metric learning and attention mechanisms to simultaneously perform intent detection and slot filling, improving contextual discrimination and interpretability.
- Triplet-loss-based metric optimization that explicitly structures the embedding space, enhancing generalization to minority intent–slot pairs and improving semantic compactness across linguistically varied utterances.
- A two-stage end-to-end SLU pipeline that bridges semantic similarity modeling and multi-task learning, coupling embedding geometry with joint decoding objectives for balanced and robust performance across both imbalanced (ATIS) and balanced (SNIPS) datasets.

Related work

Joint learning architectures

Early approaches to SLU treated intent detection and slot filling as independent tasks in a pipeline, where the output of intent classification fed directly into slot filling. Although simple, such architectures suffered from error propagation, where intent misclassification degraded slot filling performance[6]. To address this, joint learning frameworks were introduced to train both tasks simultaneously, improving contextual representation and information sharing[12].

The slot-gated model[7] used a gating mechanism to control information flow between tasks, while bi-directional joint networks[8] modeled dependencies in both directions. More recently, CEA-Net[9] incorporated co-interactive mechanism to refine intent–slot representations dynamically. Despite these advances, most of these architectures rely heavily on token-level embeddings, limiting their ability to generalize to minority intents and paraphrased utterances.

Metric learning and contrastive representation in NLU

Metric learning has emerged as an effective means of improving semantic representation robustness by organizing embedding spaces according to distance-based similarity rather than discrete classification boundaries[13]. Early studies such as LIDSNet[14] demonstrated how Siamese networks with contrastive loss enhance intent separability by modeling relationships between semantically related utterances. Building on this foundation, recent research has advanced contrastive representation learning in NLU to improve discrimination, and generalization in low-resource and cross-domain scenarios. Yang[15] combined mutual-information maximization and contrastive learning to enhance the discriminability of intent keywords and strengthen few-shot robustness. Chen[16] proposed the PFE-NBCC framework to strengthen feature extraction and clustering for new-intent discovery. Zhang[17] designed a contrastive task-adaptation model that leverages self-attention and contrastive objectives to adapt to unseen tasks, tackling the overreliance of few-shot models on base-class knowledge. Xu[18] developed a dual-level contrastive learning approach for cross-domain named-entity recognition, aimed at mitigating representation confusion between entity and non-entity tokens during domain transfer. Complementing these advances, Soleymanbaigi[19] proposed an encoder–decoder factorization model optimized with β-divergence to refine latent representations, enhance cluster separability, and improve interpretability through reconstruction consistency and manifold regularization.

These studies collectively illustrate a shift toward contrastive and metric-based representation learning, emphasizing adaptive sampling and embedding-space refinement. However, most of these works focus on representation separation for single tasks, not on joint intent–slot modeling. This motivates a more integrated approach that couples metric-based similarity learning with multi-task decoding, forming the conceptual basis of the proposed SBJLIS.

Attention and similarity fusion

Attention mechanisms enhance SLU performance by enabling models to focus on semantically important tokens. Co-interactive attention frameworks such as CEA-Net[9] and transformer-based SLU models[20] capture both local and global contextual dependencies, improving interpretability and dynamic token interactions. However, these models typically employ attention as a feature-weighting mechanism detached from any distance-based metric objective. This separation limits their ability to align token-level relevance with semantic-space geometry, which could otherwise improve compactness, interpretability, and generalization.

Despite substantial progress across joint learning, metric learning, and attention-based modeling, no prior work unifies triplet-loss-driven metric learning with attention-guided joint decoding in a single end-to-end SLU architecture. Works such as LIDSNet[14] and Yang[15] employed Siamese or contrastive encoders but focused solely on intent classification, without extending metric supervision to slot filling or joint decoding. Conversely, attention-based architectures like CEA-Net[9] improved token-level interaction but lacked embedding-space regularization, leading to overfitting in semantically overlapping or minority intents.

The proposed Siamese-Based Joint Learning for Intent Detection and Slot Filling (SBJLIS) addresses this methodological gap through a unified two-stage framework. In the first stage, triplet-based metric pretraining

shapes a semantically coherent embedding space by enforcing compactness among similar utterances and separation among dissimilar ones. In the second stage, attention-enhanced joint decoding leverages these structured embeddings to dynamically align token importance with sentence-level semantics.

This alignment between embedding geometry and joint decoding offers two key benefits. Theoretically, it couples metric geometry with joint decoding objectives. Empirically, it improves generalization under class imbalance and linguistic variation, which remain major challenges in SLU research.

Proposed methodology

This section presents the proposed SBJLIS model. The model's novelty lies primarily in its two-stage training framework, consisting of Siamese triplet-based pretraining for discriminative sentence embedding learning and attention-based joint fine-tuning for multi-task optimization.

Model overview

As shown in Fig. 1, SBJLIS begins with a triplet construction stage that generates anchor, positive, and negative utterance samples from the training dataset. These are passed through a Siamese encoder to produce discriminative sentence embeddings optimized via triplet loss. The learned embeddings are then transferred into a shared BiLSTM–attention joint decoder. During this second phase, the Siamese weights are fine-tuned jointly with the downstream network, ensuring full end-to-end optimization. See Algorithm 1 for the complete training procedure.

Triplet selection

Triplet selection is the first phase of the model pipeline, responsible for organising training data into structured triplets. Each triplet comprises:

- Anchor (A): A reference input query (e.g., *"book a flight to Boston"*)

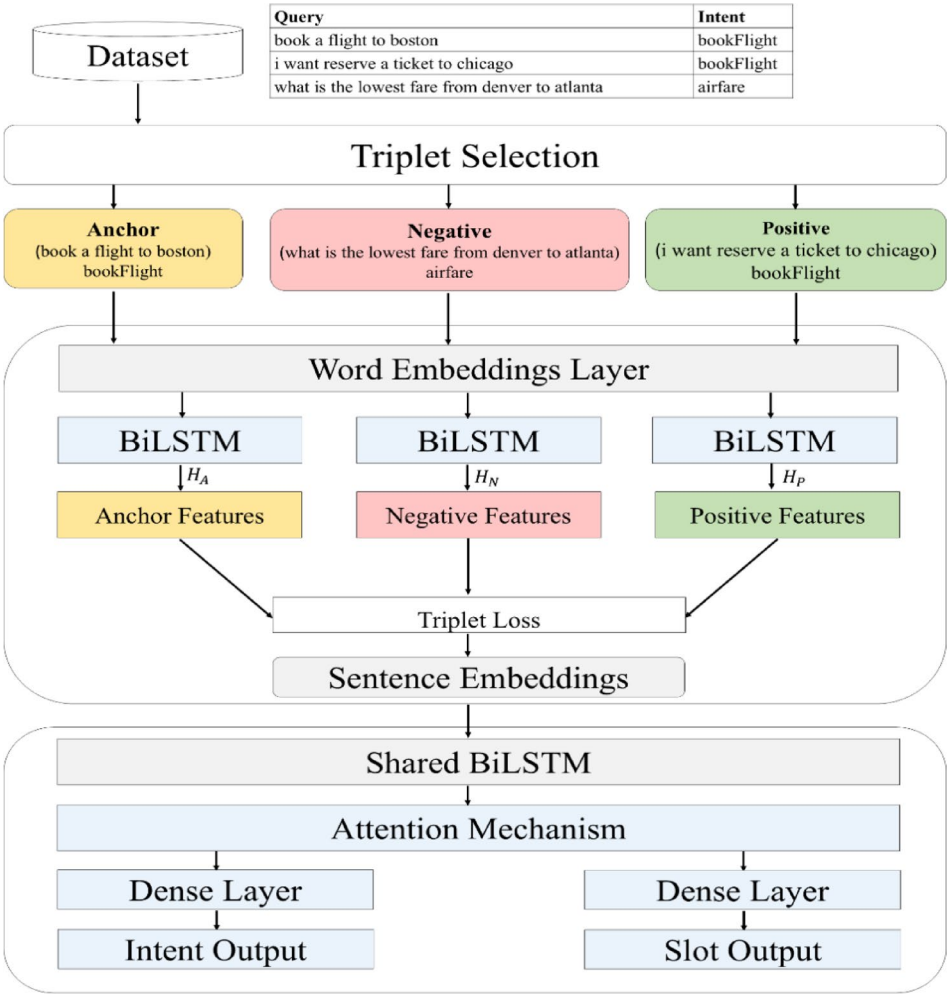


Fig. 1. SBJLIS two-stage architecture. Stage-1: a Siamese learns sentence embeddings with triplet loss using anchor–positive–negative triplets. Stage-2: the pretrained encoder initializes an attention-based joint decoder; sentence-level context supports intent classification while token-level states support slot filling.

- Positive (P): A semantically similar query from the same intent class (e.g., *"I want to reserve a ticket to Chicago"*)
- Negative (N): A semantically dissimilar query from a different intent class (e.g., *"what is the lowest fare from Denver to Atlanta"*).

This setup ensures that the anchor and positive share the same intent label, while the negative introduces semantic contrast, enabling the network to minimize intra-class distances and maximize inter-class separability. To maintain balanced class representation and stable convergence, uniform random sampling is used by selecting positives from the same intent class and negatives from different ones, which reduces class-frequency bias and ensures proportional contribution of all intents during training. This uniform random strategy serves as a baseline for future enhancements involving semi-hard or hard-negative mining approaches.

Siamese network encoder

Once the triplets are generated, the samples are fed into a Siamese network encoder consisting of BiLSTM layers. This network is trained with triplet loss to produce sentence embeddings that represent semantic proximity. The training process consists of three key steps: embedding generation, distance calculation, and loss optimization, all of which contribute to refining the model's ability to generalize across diverse user queries.

Embedding generation

The model begins with feature representation using pre-trained Word2Vec embeddings, which map each input token to a dense vector representation. These embeddings are derived from a large corpus of 100 billion words from the Google News dataset[21] and are known to capture rich semantic relationships between words.

Given an input sequence $X = (x_1, x_2, \ldots, x_T)$, the Word2Vec embedding layer transforms each token x_i into a vector $\mathbf{E}(x_i)$ resulting in the embedding matrix:

$$\boldsymbol{E} = [\boldsymbol{E}(x_1), \boldsymbol{E}(x_2), \ldots, \boldsymbol{E}(x_T)] \in \mathbb{R}^{T \times d} \tag{1}$$

where T is the sequence length and d is the embedding dimension.

These embedding vectors are then fed into the BiLSTM layer, which captures contextual dependencies in both forward and backward directions. The forward ($\overrightarrow{\boldsymbol{h}}_t$) and backward $\left(\overleftarrow{\boldsymbol{h}}_t\right)$ hidden states at each time step are computed as:

$$\overrightarrow{\boldsymbol{h}}_t = LSTM_{forward}\left(\boldsymbol{E}(x_t), \overrightarrow{\boldsymbol{h}}_{t-1}\right) \in \mathbb{R}^h \tag{2}$$

$$\overleftarrow{\boldsymbol{h}}_t = LSTM_{backward}\left(\boldsymbol{E}(x_t), \overleftarrow{\boldsymbol{h}}_{t-1}\right) \in \mathbb{R}^h \tag{3}$$

$$\boldsymbol{H}_t = \left[\overrightarrow{\boldsymbol{h}}_t; \overleftarrow{\boldsymbol{h}}_t\right] \in \mathbb{R}^{2h} \tag{4}$$

The final output $\boldsymbol{H}_t$ is the concatenation of the forward and backward hidden states at each time step, producing context-aware feature embeddings for the anchor ($\boldsymbol{H}_A$), positive ($\boldsymbol{H}_P$), and negative ($\boldsymbol{H}_N$) samples.

Distance calculation

After generating the contextual embeddings, the model computes the squared Euclidean distances between the anchor–positive and anchor–negative pairs within a shared d-dimensional semantic space:

$$d_{AP} = \|\boldsymbol{H}_A - \boldsymbol{H}_P\|_2^2 \tag{5}$$

$$d_{AN} = \|\boldsymbol{H}_A - \boldsymbol{H}_N\|_2^2 \tag{6}$$

where d_{AP} is the distance between anchor and positive and d_{AN} is the distance between anchor and negative.

Loss optimization

To enforce the desired separation between similar and dissimilar samples, the model employs the triplet loss function, which ensures that the anchor is closer to the positive than to the negative by at least a predefined margin α. The triplet loss is defined as:

$$Loss_{triplet} = \frac{1}{m}\sum_{i=1}^{m} \max(0, d_{AP} - d_{AN} + \alpha) \tag{7}$$

where m is the total number of triplets in a batch, α is the margin hyperparameter that enforces a sufficient gap between positive and negative pairs.

If the condition $d_{AP} + \alpha < d_{AN}$ is met, the loss is zero, indicating a well-learned separation. Otherwise, the loss penalizes insufficient separation, prompting the network to adjust embeddings for better class discrimination.

Attention based joint decoder for intent detection and slot filling

In this stage, the optimized semantic embeddings from the Siamese network are further processed in a multi-task learning framework. This framework consists of three components: a shared BiLSTM layer for contextual encoding, an attention mechanism to highlight salient information, and dual output heads for intent classification and slot tagging.

Shared BiLSTM layer

To capture sequential dependencies across the input, the optimized embeddings $\boldsymbol{S} = (\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_T) \in \mathbb{R}^{T \times d}$ are passed through a shared BiLSTM network. This layer models both forward and backward temporal context, producing richer token-level representations:

$$\overrightarrow{\boldsymbol{h}}_t = LSTM_{forward}\left(\boldsymbol{s}_t, \overrightarrow{\boldsymbol{h}}_{t-1}\right) \tag{8}$$

$$\overleftarrow{\boldsymbol{h}}_t = LSTM_{backward}\left(\boldsymbol{s}_t, \overleftarrow{\boldsymbol{h}}_{t-1}\right) \tag{9}$$

$$\boldsymbol{H}_t = \left[\overrightarrow{\boldsymbol{h}}_t; \overleftarrow{\boldsymbol{h}}_t\right] \in \mathbb{R}^{2h} \tag{10}$$

where $\overrightarrow{\boldsymbol{h}}_t$ and $\overleftarrow{\boldsymbol{h}}_t$ represent the hidden states from the forward and backward LSTMs at time step t, and $\boldsymbol{H}_t$ is the concatenated hidden state representing the contextualized representation of token t.

This layer serves as a common encoder for both the intent detection and slot filling tasks, enabling knowledge sharing across objectives.

Attention mechanism

To enhance the model's ability to focus on semantically important tokens, an attention mechanism is applied to the BiLSTM outputs. This mechanism enables the model to assign varying levels of importance to each token based on its relevance to the overall sentence meaning. The attention process begins by calculating a raw attention score for each token at a given time step t, denoted as e_t:

$$\boldsymbol{e}_t = tanh\left(\boldsymbol{W}_a \boldsymbol{H}_t + \boldsymbol{b}_a\right) \tag{11}$$

where $\boldsymbol{W}_a$ and $\boldsymbol{b}_a$ are learnable parameters, while $\boldsymbol{H}_t$ represents the contextual BiLSTM output at time step t.

The attention scores $\boldsymbol{e}_t$ are then normalized using the Softmax function to produce attention weights a_t, defined as:

$$a_t = \frac{exp(\boldsymbol{e}_t)}{\sum_{k=1}^{T} exp(\boldsymbol{e}_k)} \tag{12}$$

These weights quantify the relative contribution of each token to the overall sentence representation. Subsequently, the model computes a context vector $\boldsymbol{c}$ as a weighted sum of all BiLSTM hidden states, using the formula:

$$\boldsymbol{c} = \sum_{t=1}^{T} a_t \boldsymbol{H}_t \epsilon \mathbb{R}^{2h} \tag{13}$$

The resulting vector $\boldsymbol{c}$ serves as a global summary of the input sequence, selectively emphasizing more informative tokens based on their attention scores. This contributes to the downstream intent classification task by providing a semantically rich and focused representation of the entire sentence.

However, for the slot filling task, instead of using the sentence-level context vector $\boldsymbol{c}$, a token-wise granularity is used by assigning an attention weight a_t to each token representation $\boldsymbol{H}_t$, producing a refined hidden state $\left(\widetilde{\boldsymbol{H}}_t\right)$ for each token:

$$\widetilde{\boldsymbol{H}}_t = a_t \boldsymbol{H}_t \tag{14}$$

Separating sentence-level and token-level representations allows the model to handle information at different semantic granularities. The sentence-level context vector captures global intent semantics, providing holistic understanding of the utterance, while the token-level states preserve fine-grained contextual cues essential for slot labeling. This distinction prevents interference between the two objectives, enabling the model to optimize intent and slot predictions more effectively and improving both accuracy and interpretability.

Multitask output

To predict the user's intent, the attention-derived sentence vector $\boldsymbol{c}$, which encapsulates the global semantics of the input sequence, is processed through a Softmax classifier.

$$\boldsymbol{y}^i = Softmax\left(\boldsymbol{W}_i \boldsymbol{c} + \boldsymbol{b}_i\right) \tag{15}$$

where $\boldsymbol{W}_i$, and $\boldsymbol{b}_i$ are the learnable parameters, and $\boldsymbol{y}^i$ is the predicted distribution over possible intent classes.

For slot filling, the attention refined token vector $\widetilde{\boldsymbol{H}}_t$ is passed through a Softmax layer for slot filling:

$$\boldsymbol{y}^s = SoftMax\left(\boldsymbol{W}_s \bullet \widetilde{\boldsymbol{H}}_t + \boldsymbol{b}_s\right) \tag{16}$$

where $\boldsymbol{W}_s$, and $\boldsymbol{b}_s$ are learnable parameters, and $\boldsymbol{y}^s$ is the probability distribution over slot labels for token t.

The model jointly optimizes both intent detection and slot filling under a unified multi-task learning objective expressed as:

$$p\left(\boldsymbol{y}^i, \boldsymbol{y}^s|x\right) = p\left(\boldsymbol{y}^i|x\right) \cdot \prod_{t=1}^{n} p\left(\boldsymbol{y}_t^s|x\right) \tag{17}$$

where $p\left(\boldsymbol{y}^i|x\right)$ is the probability of predicting the correct intent given x, and $p\left(\boldsymbol{y}_t^s|x\right)$ is the probability of predicting the correct slot label for token t given x.

This joint optimization encourages the model to share and reuse meaningful linguistic features across both tasks, thereby enhancing overall accuracy, and generalization capability. The training process of SBJLIS, encompassing both the Siamese metric training and the attention-based joint fine-tuning stages, is summarized in Algorithm 1.

Input:

- Training Dataset D containing utterances $D = \{utterances,\ labels\}$

Output: Trained SBJLIS model

Stage 1: Siamese network training

1. Construct triplets (A, P, N) from D via uniform random sampling.
2. Encode A, P, N with a shared BiLSTM to obtain $\mathbf{H}_A, \mathbf{H}_P, \mathbf{H}_N$.
3. Compute distances d_{AP}, d_{AN} and minimize $\mathcal{L}_{\text{triplet}}$.
4. Save pretrained Siamese encoder weights

Stage 2: Attention-based joint fine-tuning

5. Initialize joint model with pretrained Siamese encoder weights
6. Add attention; derive $\mathbf{c}$(intent) and $\widetilde{\mathbf{H}}_t$(slots).
7. Optimize total loss
8. Return fine-tuned SBJLIS model

Algorithm 1 Two-stage training procedure for SBJLIS.

Experimental setup

This section details the experimental settings used to evaluate the effectiveness of the proposed SBJLIS. The setup includes dataset description, preprocessing, hyperparameter configuration, baseline comparisons, computational cost, and evaluation metrics.

Datasets and preprocessing

The proposed SBJLIS model was evaluated on two widely recognized benchmark datasets for SLU: ATIS and SNIPS. These datasets provide complementary characteristics, with ATIS representing a domain-specific and imbalanced corpus, while SNIPS offers multi-domain.

1. *ATIS*—A domain-specific dataset containing queries related to flight reservations. It is heavily imbalanced, with approximately 75% of intent samples belonging to the *atis_flight* category. This imbalance presents challenges in generalization and learning effective class distributions.
2. *SNIPS*—A multi-domain dataset covering diverse user queries related to music, weather, restaurant bookings, and creative works. It features a more balanced intent distribution and varied linguistic expressions, making it suitable for assessing the model's generalization capability.

Table 1 summarizes the key statistics of the ATIS and SNIPS datasets, including the number of intents, slot labels, and the sizes of the training, test, and validation sets.

Both datasets were used in their standard cleaned form, containing only lowercase tokens and no special characters. Utterances were tokenized using the Keras Tokenizer, converting words into integer sequences based on the training vocabulary. Out-of-vocabulary words were mapped to a reserved index. Sequences were post-padded to the maximum observed lengths (46 tokens for ATIS and 35 for SNIPS) to ensure uniform input dimensions. Intent labels were one-hot encoded, while slot annotations followed the Inside–Outside–Beginning (IOB) tagging format. Slot sequences were padded to align with tokenized utterances, maintaining sequence integrity.

Hyperparameters

Hyperparameter optimization was performed through grid search and empirical tuning to ensure stable convergence and strong generalization across datasets. Each parameter range was selected based on prior studies in deep learning for natural language understanding[7,22,23] and validated experimentally on the ATIS and SNIPS datasets. The final configurations and their rationale are presented below.

1. *Triplet margin:* Values in the range [0.1–1.0] are tested, consistent with standard metric-learning practices where normalized embeddings typically exhibit Euclidean distances within [0, 2]. A margin of 0.1 achieves the most stable convergence, providing the best balance between intra-class compactness and inter-class separability.
2. *Batch size:* Batch sizes from 16 to 512 are evaluated to balance gradient diversity and convergence stability. A smaller batch size of 16 performs best on the imbalanced ATIS dataset, as higher gradient noise improves exposure to minority intents. In contrast, the balanced SNIPS dataset performs optimally with a batch size of 512, which stabilizes gradient updates and enhances training efficiency.
3. *Dropout rate:* Dropout rates between 0.1 and 0.5 are explored. A 0.3 dropout rate is applied after the embedding layer to regularize feature extraction, while 0.5 dropout is applied after the shared BiLSTM layer to prevent neuron co-adaptation. These values yield the best validation performance without hindering convergence.
4. *BiLSTM hidden state size:* A grid search across 64 to 256 hidden units identifies 128 units per direction as the optimal size for the Siamese stage, balancing computational efficiency and representational strength. In the joint learning stage, 200 units per direction further enrich contextual representation without overfitting, ensuring adequate depth for both sentence-level and token-level learning.
5. *Regularization:* L2 weight regularization is applied to dense layers. Coefficients between 0.0001 and 0.01 are examined, with 0.01 yielding the best trade-off between complexity control and performance stability.
6. *Optimizer and learning rate:* The Adam optimizer is adopted for its adaptive learning capability. Learning rates between 0.0001 and 0.01 are tested, and 0.001 provides the most stable and consistent convergence across experiments.
7. *Epochs and early stopping:* All models are trained for 10 epochs with early stopping based on validation loss. In preliminary trials, convergence typically occurred within 8 epochs on ATIS and 10 on SNIPS.
8. *Number of runs and statistical reliability:* Each experiment is repeated five times, and results are reported as the mean ± standard deviation to reduce the influence of initialization variance and confirm statistical robustness.

Dataset	No. intents	No. slots	Training set	Test set	Validation set
ATIS	21	128	4478	893	500
SNIPS	7	72	13,084	700	700

Table 1. Summary of the ATIS and SNIPS datasets used for model evaluation.

The model is implemented in Python 3.11.5 using TensorFlow 2.15.0 and Keras 2.15.0. All experiments are conducted on a Windows 10 environment with an Intel Core i7 (3.0 GHz) processor and 16 GB RAM. SBJLIS remains lightweight, comprising 2.25 million parameters (8.6 MB) and achieving an average inference time of 5.8 ms per utterance, demonstrating its computational efficiency and suitability for real-time SLU applications.

Baseline models

Although Siamese networks have been applied to intent detection in prior studies, to the best of our knowledge, no existing work has extended this architecture to the joint learning of intent detection and slot filling. This study addresses that gap by adapting the Siamese framework to simultaneously perform both tasks within a unified model.

For comparative evaluation, the proposed SBJLIS model is benchmarked against the following baseline models:

1. *Slot-Gated*[7]—Introduces a slot-gated mechanism to capture and leverage the relationship between intent detection and slot filling, thereby enhancing the performance of both tasks.
2. *LIDSNet*[14]—Employs a deep Siamese network to learn sentence representations for intent detection. It achieves competitive accuracy with a relatively small model size, making it suitable for on-device deployment.
3. *Bi-directional joint learning model*[8]—Utilizes a bidirectional inter-task learning mechanism to enhance mutual performance between intent classification and slot filling, yielding improved accuracy on benchmark datasets.
4. *CTRAN*[20]—An encoder–decoder architecture that integrates BERT embeddings, convolutional layers, and Transformer blocks for joint intent detection and slot filling.
5. *CEA-Net*[9]—Incorporates a co-interactive external attention mechanism to capture complex interactions between intent and slot representations, improving spoken language understanding accuracy.

Evaluation metrics

The evaluation of the proposed SBJLIS model aligns with its two-phase architecture, with each phase using metrics suited to its objectives.

Phase I—Metric evaluation for Siamese network

In the first phase, the Siamese network is trained using triplet loss, which encourages embeddings of semantically similar utterances to be close while pushing apart those of dissimilar utterances.

Performance at this stage is monitored using the following metrics:

- *Training loss*—The average triplet loss computed over each training batch.
- *Validation loss*—The triplet loss computed on a held-out validation set to assess embedding generalization.

Both training and validation losses were recorded across different margin values and batch sizes to determine optimal configurations for semantic separation.

Phase II—task-level evaluation for joint learning

In the second phase, the learned sentence-level embeddings are fed into the joint BiLSTM–attention model for simultaneous intent detection and slot filling.

Intent detection accuracy For intent detection, the model's performance is evaluated using accuracy, defined as:

$$Accuracy = \frac{No.\ of\ correct\ intent\ prediction}{No.\ of\ utterances} \tag{18}$$

Slot filling—span-based micro-averaged F1-score For slot filling, the span-based micro-averaged F1-score is reported, following the standard evaluation protocol in SLU literature[9,24–26]. In this approach, a predicted slot is considered correct only if both the slot type and the exact span boundaries match the ground truth. This avoids inflated scores from partial token matches and ensures direct comparability with prior work.

The span-based micro-averaged F1-score is computed by first aggregating the true positives, false positives, and false negatives across all slot types and then calculating the overall precision, recall, and F1-score as follows:

$$Precision = \frac{True\ positives}{True\ positives + False\ positives} \tag{19}$$

$$Recall = \frac{True\ positives}{True\ positives + False\ negatives} \tag{20}$$

$$F1 - score = \frac{Precision.Recall}{Precision + Recall} \tag{21}$$

Here, true positives refer to correctly predicted slot spans, false positives to predicted spans not in the ground truth, and false negatives to ground truth spans missed by the model. The use of span-based evaluation ensures a rigorous and fair assessment of the slot filling task, particularly in datasets containing multi-token slot entities.

	ATIS dataset											
	Batch size											
	16		32		64		128		256		512	
Margin	Loss	Val_loss	Loss	Val_loss	Loss	Val_loss	Loss	Val_loss	Loss	Val_loss	Loss	Val_loss
0.1	0.0073	0.0151	0.0065	0.0151	0.0065	0.0151	0.0065	0.0152	0.0078	0.0153	0.0095	0.0158
0.2	0.0111	0.0220	0.0130	0.0217	0.0115	0.0262	0.0132	0.0226	0.0170	0.0259	0.0277	0.0307
0.3	0.0153	0.0255	0.0145	0.0332	0.0154	0.0287	0.0193	0.0291	0.0256	0.0337	0.0427	0.0428
0.4	0.0180	0.0305	0.0193	0.0343	0.0210	0.0341	0.0275	0.0378	0.0352	0.0459	0.0589	0.0572
0.5	0.0208	0.0399	0.0223	0.0372	0.0273	0.0454	0.0341	0.0438	0.0448	0.0543	0.0783	0.0729
0.6	0.0260	0.0387	0.0256	0.0388	0.0316	0.0401	0.0416	0.0530	0.0564	0.0706	0.0971	0.0882
0.7	0.0311	0.0460	0.0314	0.0503	0.0369	0.0483	0.0482	0.0557	0.0674	0.0743	0.1181	0.1037
0.8	0.0347	0.0453	0.0348	0.0683	0.0424	0.0545	0.0572	0.0693	0.0797	0.0892	0.1398	0.1210
0.9	0.0413	0.0490	0.0393	0.0729	0.0472	0.0639	0.0690	0.0733	0.0905	0.0948	0.1612	0.1381
1.0	0.0478	0.0614	0.0376	0.0647	0.0427	0.0631	0.0766	0.0820	0.1073	0.1223	0.2226	0.1459

Table 2. Training and validation loss for Siamese network on ATIS dataset.

	SNIPS dataset											
	Batch size											
	16		32		64		128		256		512	
Margin	Loss	Val_loss	Loss	Val_loss	Loss	Val_loss	Loss	Val_loss	Loss	Val_loss	Loss	Val_loss
0.1	0.00221	0.00628	0.00200	0.00846	0.00200	0.00702	0.00191	0.00552	0.00227	0.00645	0.00336	0.00449
0.2	0.00477	0.01254	0.00297	0.01215	0.00322	0.01166	0.00309	0.01009	0.00393	0.01025	0.00591	0.00715
0.3	0.00475	0.01875	0.00388	0.01533	0.00428	0.01792	0.00453	0.01316	0.00606	0.01420	0.00913	0.01029
0.4	0.00597	0.01522	0.00495	0.01599	0.00557	0.01910	0.00580	0.01702	0.00822	0.01697	0.01270	0.01127
0.5	0.00680	0.02096	0.00605	0.02143	0.00681	0.02240	0.00755	0.02079	0.00845	0.01958	0.01688	0.01218
0.6	0.00721	0.02708	0.00675	0.02571	0.00819	0.02684	0.00941	0.02312	0.00854	0.02478	0.02137	0.01811
0.7	0.00886	0.02947	0.00777	0.02509	0.00928	0.02515	0.01093	0.02578	0.02428	0.01960	0.02428	0.01960
0.8	0.01032	0.03339	0.00923	0.02839	0.01114	0.03364	0.01325	0.02857	0.01970	0.02727	0.02878	0.02662
0.9	0.01202	0.03280	0.01015	0.02753	0.01273	0.04063	0.01461	0.03074	0.02275	0.03720	0.03650	0.02769
1.0	0.01162	0.03805	0.01176	0.03281	0.01256	0.0405	0.01492	0.03164	0.02595	0.04321	0.04175	0.02889

Table 3. Training and validation loss for Siamese network on SNIPS dataset.

Results and discussion

The results are presented in two stages, reflecting the two-phase training strategy of SBJLIS: Siamese network embedding learning, and attention-based joint decoder using the learned embeddings.

Siamese network embedding learning

Impact of triplet loss margin

The triplet margin controls separation between dissimilar samples and therefore shapes the embedding geometry. Tables 2 and 3 show clear patterns. On ATIS, small margins (0.1–0.2) minimize validation loss and keep train–val gaps small. For example, at margin 0.1 with batch 16, losses are 0.0073 and 0.0151, which indicates good generalization. As the margin exceeds 0.4, the gap widens; at margin 1.0 with batch 512, validation loss rises to 0.1459. Consequently, large margins over-separate classes.

SNIPS exhibits the same trend. Margins of 0.1–0.2 yield the lowest validation losses (down to 0.00449). In contrast, margins above 0.4 degrade learning because strict constraints hinder smooth transitions among semantically related intents.

These results show that smaller margins create compact yet flexible embeddings that preserve local semantic continuity, improving generalization. Conversely, larger margins push samples too far apart, breaking semantic continuity and limiting the model's ability to capture related intents, especially in minority classes. This aligns with metric learning theory, where moderate margins promote an optimal equilibrium between intra-class cohesion and inter-class separation, supporting adaptive discrimination across variable data distributions[22].

Effect of batch size on performance

Batch size plays a central role in determining the stability and generalization of neural models. It influences gradient noise, convergence speed, and embedding compactness. Examining its effect reveals how training behavior adapts to dataset balance and class overlap, particularly under the optimal triplet margin configuration (0.1).

For ATIS, validation loss remains nearly constant (~0.015) across batch sizes at margin 0.1, showing that its domain-specific and imbalanced data require minimal gradient stabilization. However, larger batches combined with higher margins (≥0.5) induce overfitting, as reflected by a sharp increase in validation loss (from 0.0614 at batch 16 to 0.1459 at batch 512). Smaller batches (16–64) perform better by introducing beneficial stochasticity during optimization, which helps capture minority intents. Conversely, SNIPS benefits from larger batches due to its balanced and multi-domain nature. At margin 0.1, validation loss decreases from 0.00628 (batch 16) to 0.00449 (batch 512), signifying smoother gradient updates and more stable embedding learning.

The t-SNE visualizations in Figs. 2 and 3 clearly illustrate these patterns. In ATIS (Fig. 2), embeddings remain moderately compact across all batch sizes, but larger batches (≥256) show tighter, rigid clusters that risk over-separation. Smaller batches exhibit slight overlap among clusters, improving the representation of minority intents.

In SNIPS (Fig. 3), as batch size increases from 16 to 512, clusters become progressively cleaner and more distinct, reflecting enhanced inter-intent separation and reduced ambiguity among overlapping classes. These structural differences align with the quantitative findings: smaller batches favor diversity and regularization in imbalanced corpora, while larger batches stabilize training for balanced, linguistically varied datasets.

These results show that the interaction between batch size and dataset distribution determines embedding cohesion and generalization. Smaller batches enhance robustness on skewed datasets like ATIS, whereas larger batches yield superior structure and convergence on balanced datasets such as SNIPS. Aligning batch configuration with dataset characteristics thus ensures efficient learning and optimal embedding geometry.

Siamese-based joint model for intent detection and slot filling

This section evaluates the effectiveness of the sentence embeddings generated by the Siamese network when integrated into the attention-based joint learning model. The embeddings serve as inputs to an attention mechanism that dynamically assigns token importance, allowing the model to focus on the most informative words for both intent detection and slot filling. All reported results represent the mean ± standard deviation over five independent runs to ensure statistical reliability.

The evaluation proceeds in two stages. First, the model is trained and tested with embeddings produced under different triplet loss margins to observe their influence on class separability and downstream performance. Second, the best-performing margin configuration is re-evaluated across varying batch sizes to examine how training stability and generalization respond to different gradient update conditions. Together, these analyses reveal the interaction between metric-based pretraining, attention-driven decoding, and dataset characteristics.

Table 4 and Fig. 4 summarize the performance of SBJLIS across different triplet loss margins. Small margins (0.1–0.2) consistently yield the highest accuracy and F1-scores for both datasets. For ATIS, a margin of 0.1 achieves 98.87 ± 0.02% accuracy and 98.60 ± 0.04% F1-score, while SNIPS reaches 99.23 ± 0.01% accuracy and 98.43 ± 0.03% F1-score. Larger margins (>0.4) progressively reduce performance as embeddings become excessively separated, weakening semantic cohesion among related intents. Conversely, smaller margins preserve local relationships, allowing the attention layer to capture subtle contextual overlaps between tokens. The low standard deviations further confirm the model's stable convergence and repeatability across runs.

The impact of batch size was then analyzed using the best-performing margin setting (0.1). Results in Table 5 show that for ATIS, smaller batches (16–32) produce superior accuracy and F1-scores, reaching 98.87 ± 0.02% and 98.60 ± 0.04%, because frequent updates increase exposure to minority classes in the imbalanced dataset. In contrast, SNIPS benefits from larger batches (128–512), which improve gradient stability and convergence, achieving 99.61 ± 0.02% accuracy and 98.68 ± 0.03% F1-score. These outcomes demonstrate how dataset balance dictates the optimal batch configuration: smaller batches introduce beneficial gradient noise in skewed data, while larger batches support consistent learning in balanced corpora.

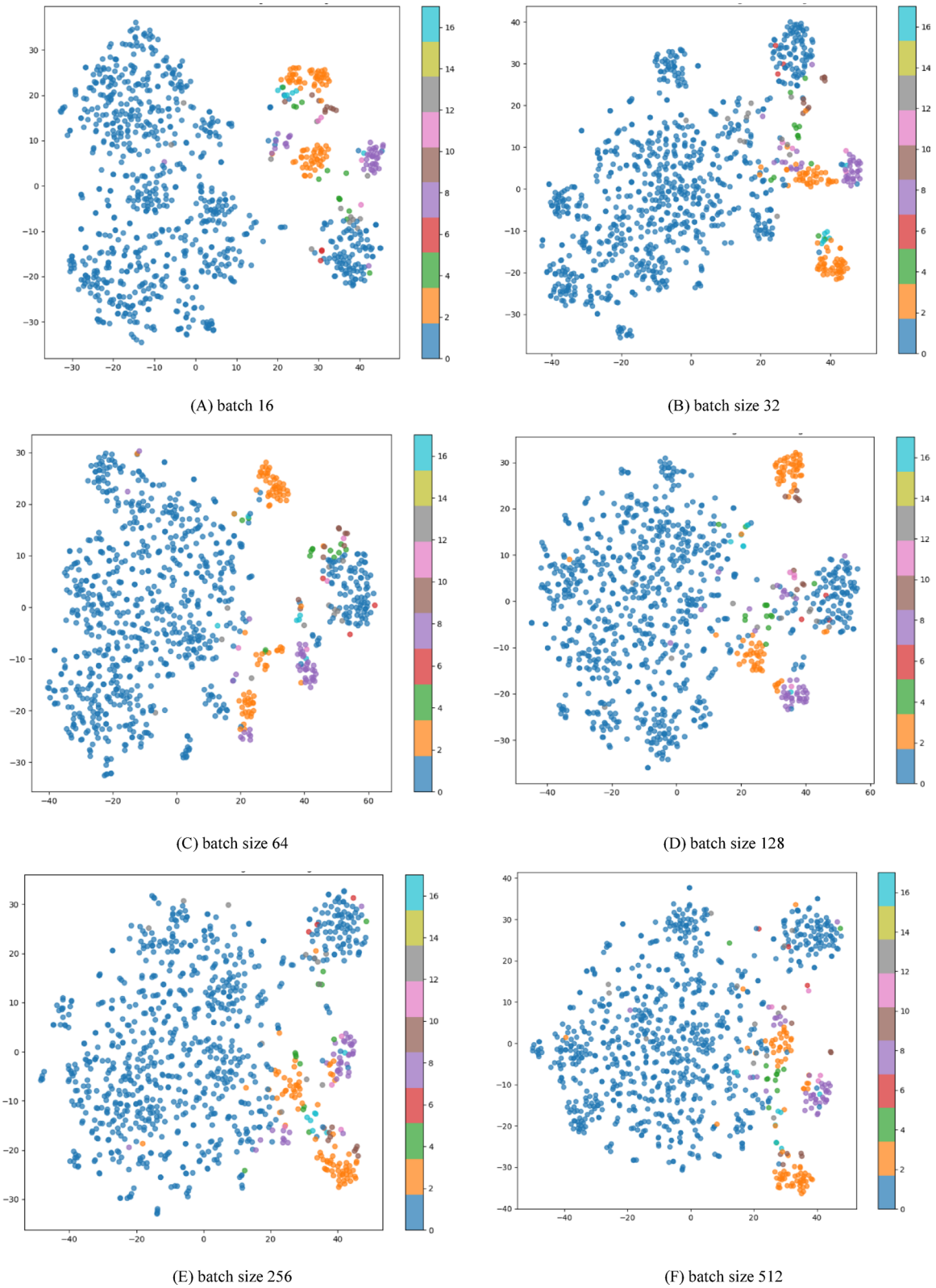
The t-SNE visualizations in Figs. 2 and 3 further illustrate these trends. In ATIS, smaller batches form overlapping but semantically meaningful clusters that preserve minority-intent structures, while very large batches compress the embedding space and reduce intra-class variation. For SNIPS, larger batches produce cleaner and more distinct clusters, confirming stable representation learning under balanced distributions. These visual results align closely with the quantitative findings, showing that optimal embedding compactness and separation depend on both dataset structure and training configuration.

The results show that SBJLIS achieves its best results when small triplet margins (0.1–0.2) are paired with dataset-specific batch sizes. These settings generate compact yet discriminative embeddings that enable the attention layer to model intent–slot dependencies more effectively. Performance gains arise from the synergy between metric-based pretraining, which enhances feature separability, and attention-based fine-tuning, which sharpens token relevance during decoding. Performance declines under large margins or poorly tuned batch configurations, where embeddings become rigid and less responsive to contextual variation.

Comparison with baseline

Table 6 and Fig. 5 present the comparison between the proposed SBJLIS model and five baseline models on the ATIS and SNIPS datasets. SBJLIS improves accuracy by ≈ 1.7% and F1-score by ≈ 2.8% on average. The model achieves the highest results across all settings, showing consistent performance on both imbalanced and balanced datasets. These gains come from the combined effect of Siamese metric learning and attention-based joint decoding, which enhance contextual discrimination and preserves semantic precision for intent detection and slot filling.

Paired two-tailed t-tests at a 95% confidence level confirm the statistical reliability of these improvements. The results in Tables 7, 8, 9 and 10 show detailed outcomes for each dataset and metric. Tables 7 and 8 report ATIS results for accuracy and F1-score, while Tables 9 and 10 present the same for SNIPS. All p-values are below 0.05, and the confidence intervals are narrow. The mean performance differences range between 1 and 3



(A) batch 16

(B) batch size 32

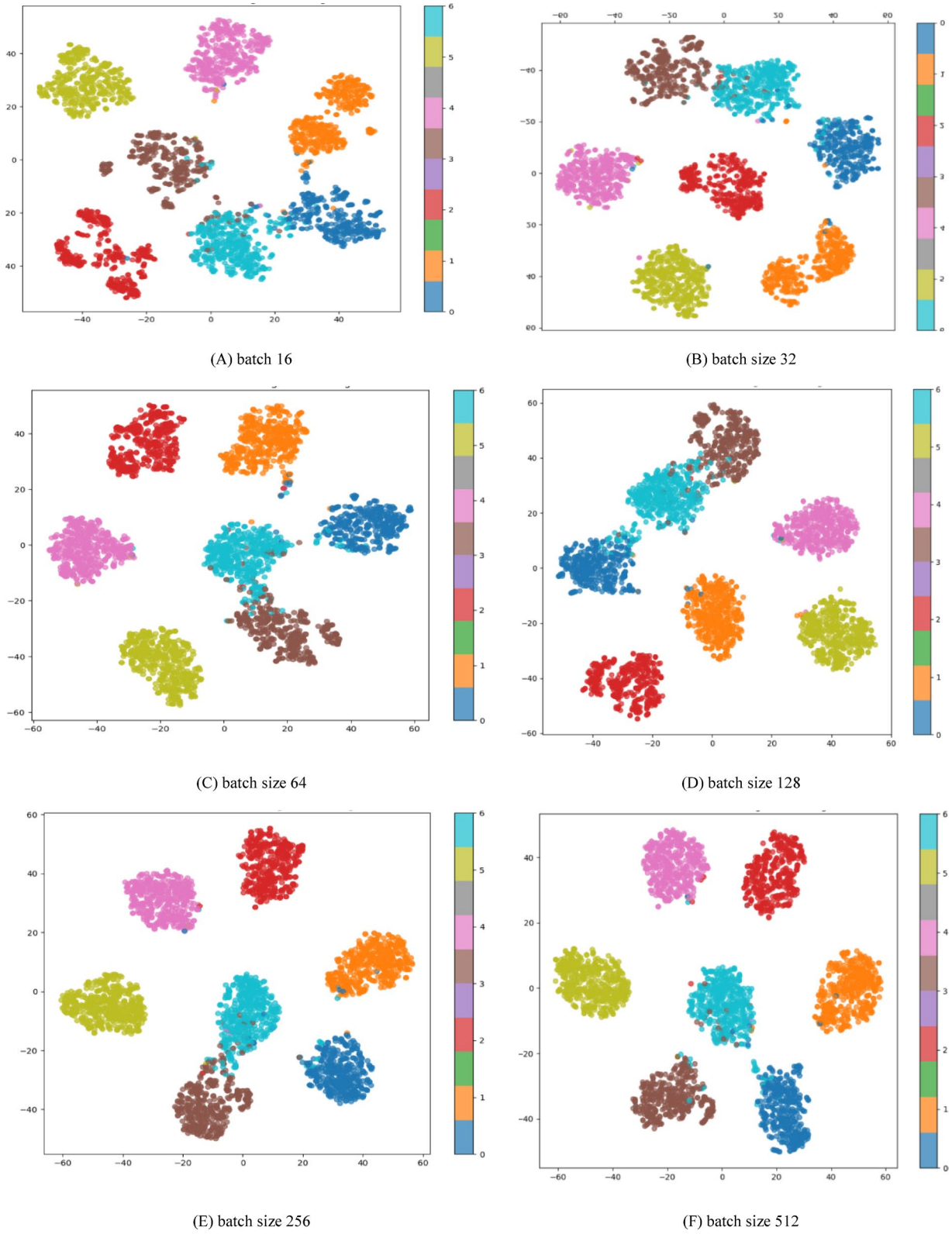
(C) batch size 64

(D) batch size 128

(E) batch size 256

(F) batch size 512

Fig. 2. t-SNE visualization of sentence embeddings for different batch sizes (16–512) on the ATIS dataset at margin 0.1. Smaller batches generate moderately overlapping clusters that preserve minority intent representation.



(A) batch 16

(B) batch size 32

(C) batch size 64

(D) batch size 128

(E) batch size 256

(F) batch size 512

Fig. 3. t-SNE visualization of sentence embeddings for different batch sizes (16–512) on the SNIPS dataset at margin 0.1. As batch size increases, clusters become cleaner and more distinct, demonstrating improved inter-intent discrimination and reduced semantic overlap in balanced multi-domain data.

Models	Margin	ATIS		SNIPS	
		Accuracy (%)	F1-score (%)	Accuracy (%)	F1-score (%)
SBJLIS_1	0.1	**98.87 ± 0.02**	**98.60 ± 0.04**	**99.23 ± 0.01**	**98.43 ± 0.03**
SBJLIS_2	0.2	97.74 ± 0.03	97.92 ± 0.05	99.00 ± 0.02	98.02 ± 0.03
SBJLIS_3	0.3	96.74 ± 0.04	98.05 ± 0.04	98.95 ± 0.02	98.30 ± 0.03
SBJLIS_4	0.4	97.11 ± 0.03	98.13 ± 0.05	98.93 ± 0.03	97.86 ± 0.04
SBJLIS_5	0.5	96.86 ± 0.04	98.43 ± 0.05	98.89 ± 0.03	97.83 ± 0.04
SBJLIS_6	0.6	96.49 ± 0.03	97.66 ± 0.05	98.58 ± 0.04	97.82 ± 0.03
SBJLIS_7	0.7	97.11 ± 0.02	98.06 ± 0.04	98.50 ± 0.03	97.81 ± 0.04
SBJLIS_8	0.8	96.74 ± 0.04	98.30 ± 0.05	98.20 ± 0.04	97.29 ± 0.04
SBJLIS_9	0.9	95.36 ± 0.05	97.73 ± 0.06	98.07 ± 0.04	97.86 ± 0.04
SBJLIS_10	1.0	96.49 ± 0.05	98.21 ± 0.05	97.63 ± 0.04	97.85 ± 0.05

Table 4. Performance of the attention-based joint learning model with Siamese network extracted embeddings on ATIS and SNIPS datasets across different triplet-loss margins (mean ± standard deviation over five runs). It shows the highest recorded performance of our model.

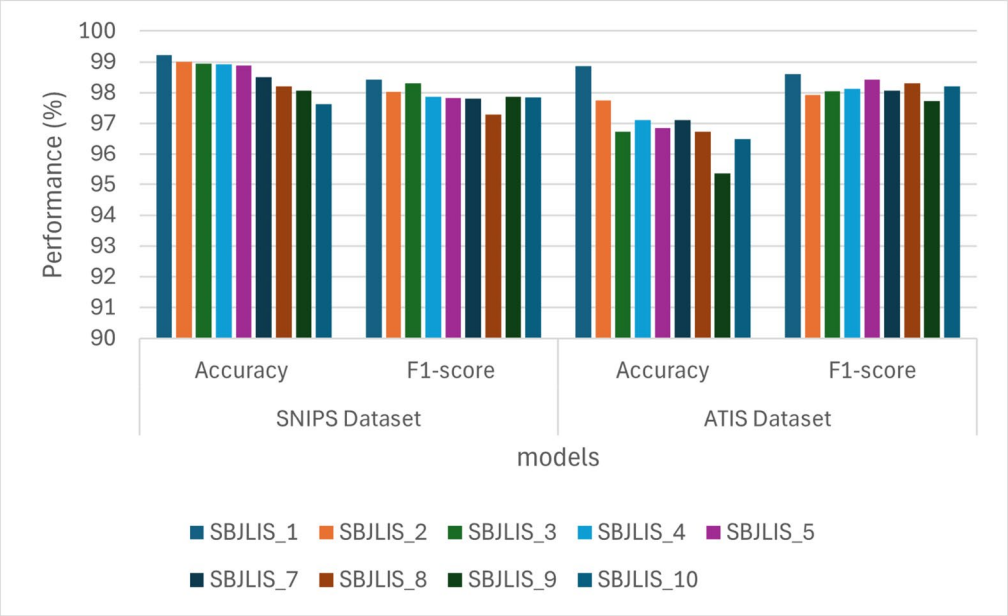


Fig. 4. Performance comparison of SBJLIS models with varying triplet-loss margins (0.1–1.0) across SNIPS and ATIS datasets. The figure shows that smaller margins (0.1–0.2) yield the highest accuracy and F1-scores, reflecting compact yet generalizable embeddings.

Models	Batch_size	ATIS		SNIPS	
		Accuracy (%)	F1-score (%)	Accuracy (%)	F1-score (%)
SBJLIS_1	16	**98.87 ± 0.02**	**98.60 ± 0.04**	99.23 ± 0.01	98.43 ± 0.03
SBJLIS_1	32	98.87 ± 0.03	97.93 ± 0.04	99.37 ± 0.02	98.47 ± 0.03
SBJLIS_1	64	98.43 ± 0.04	97.90 ± 0.05	99.34 ± 0.03	97.72 ± 0.04
SBJLIS_1	128	98.24 ± 0.04	97.61 ± 0.04	99.61 ± 0.02	98.50 ± 0.03
SBJLIS_1	256	97.99 ± 0.05	97.72 ± 0.05	99.57 ± 0.02	98.49 ± 0.03
SBJLIS_1	512	97.71 ± 0.04	97.49 ± 0.05	**99.61 ± 0.02**	**98.68 ± 0.03**

Table 5. Effect of batch size on the best-performing model (SBJLIS_1) using a margin of 0.1 (mean ± standard deviation over five runs). It shows the highest recorded performance of our model.

Baseline Models	ATIS		SNIPS	
	Accuracy (%)	F1-score (%)	Accuracy (%)	F1-score (%)
Slot-Gated[7]	94.10	95.20	97.0	88.80
LIDSNet[14]	95.97	–	98.00	–
Bi-directional[8]	98.60	96.30	99.20	97.20
CTRAN[20]	98.07	98.46	99.13	98.30
CEA-Net[9]	97.76	96.21	98.86	96.21
SBJLIS	**98.87**	**98.60**	**99.61**	**98.68**

Table 6. Performance comparison of SBJLIS with baselines. It shows the highest recorded performance of our model.

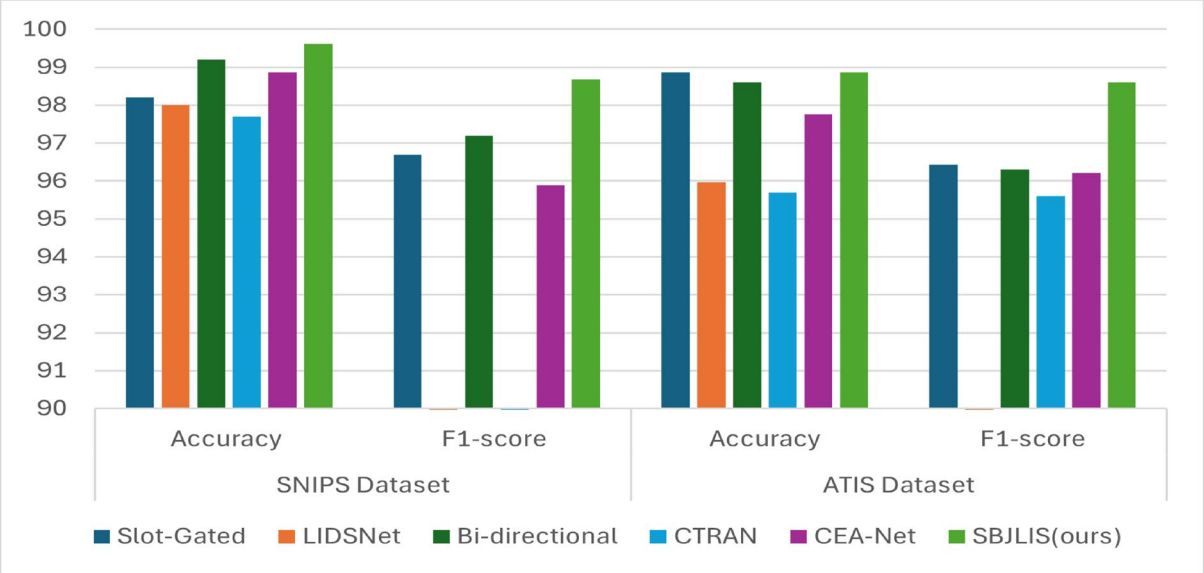


Fig. 5. Comparative performance of SBJLIS against baseline models on ATIS and SNIPS datasets. SBJLIS consistently achieves higher accuracy and F1-scores, confirming the advantage of integrating Siamese metric learning with attention-guided joint decoding for robust intent detection and slot filling.

Baseline Model	Mean Diff	Std. Dev	95% CI (Lower–Upper)	t	df	*p*
Slot-Gated	0.010	0.007	[0.001, 0.019]	3.16	4	0.034
LIDSNet	2.900	0.102	[2.773, 3.027]	63.28	4	< 0.001
Bi-directional	0.270	0.067	[0.187, 0.353]	9.05	4	< 0.001
CTRAN	3.170	0.107	[3.037, 3.302]	66.39	4	< 0.001
CEA-Net	1.110	0.107	[0.977, 1.242]	23.25	4	< 0.001

Table 7. Paired samples t-test results comparing SBJLIS with baseline models on the ATIS dataset in terms of accuracy.

Baseline model	Mean Diff	Std. Dev	95% CI (Lower–Upper)	t	df	*p*
Slot-Gated	2.580	0.021	[2.553, 2.606]	271.96	4	< 0.001
Bi-directional	2.710	0.036	[2.666, 2.754]	171.40	4	< 0.001
CTRAN	3.410	0.036	[3.366, 3.453]	215.67	4	< 0.001
CEA-Net	2.800	0.050	[2.739, 2.861]	126.49	4	< 0.001

Table 8. Paired samples t-test results comparing SBJLIS with baseline models on the ATIS dataset in terms of F1-score.

Baseline model	Mean Diff	Std. Dev	95% CI (Lower–Upper)	t	df	*p*
Slot-Gated	1.400	0.007	[1.391, 1.409]	442.72	4	< 0.001
LIDSNet	1.610	0.007	[1.601, 1.618]	509.13	4	< 0.001
Bi-directional	0.410	0.007	[0.401, 0.419]	129.65	4	< 0.001
CTRAN	1.910	0.007	[1.901, 1.918]	603.99	4	< 0.001
CEA-Net	0.750	0.007	[0.741, 0.758]	237.17	4	< 0.001

Table 9. Paired samples t-Test results comparing SBJLIS with baseline models on the SNIPS dataset in terms of accuracy.

Baseline Model	Mean Diff	Std. Dev	95% CI (Lower–Upper)	t	df	*p*
Slot-Gated	1.990	0.007	[1.981, 1.999]	629.29	4	< 0.001
Bi-directional	1.480	0.007	[1.471, 1.489]	468.02	4	< 0.001
CTRAN	9.480	0.007	[9.471, 9.489]	2997.84	4	< 0.001
CEA-Net	2.780	0.007	[2.771, 2.789]	879.11	4	< 0.001

Table 10. Paired samples t-test results comparing SBJLIS with baseline models on the SNIPS dataset in terms of F1-score.

percentage points for accuracy and 1.5–3 for F1-score. Standard deviations remain low, below 0.1 in all cases, confirming consistent behavior across multiple runs.

These results show that SBJLIS is statistically validated and reliable. The model's Siamese pretraining improves feature separability, and its attention fine-tuning enhances interpretability and generalization. The improvements across both datasets demonstrate that SBJLIS is a stable and scalable framework for joint intent detection and slot filling in natural-language-understanding tasks.

Conclusion

This study presents SBJLIS, a unified framework that integrates metric-based semantic similarity learning with attention-based joint decoding, delivering robust gains across both balanced and imbalanced SLU settings. Through a two-phase training process combining embedding learning via triplet loss and attention-based joint classification, SBJLIS addresses key limitations of existing models, namely poor generalization to rare classes and insufficiently discriminative sentence-level embeddings. Experimental results across ATIS and SNIPS demonstrate that triplet margins of 0.1–0.2 produce the most compact and generalizable embeddings. However, the optimal batch configuration varies with dataset characteristics: smaller batches (16–32) enhance robustness in the imbalanced ATIS dataset, while larger batches (128–512) improve stability and generalization in the balanced SNIPS corpus. The model's consistent outperformance of baseline approaches across both datasets demonstrates its robustness, adaptability, and real-world applicability. Future work will explore adaptive triplet mining to improve hard-example selection, extend the framework to multilingual and cross-domain spoken language understanding, and incorporate transformer-based encoders to further strengthen contextual representation. The proposed SBJLIS framework can also be deployed in practical systems, where reliable intent detection and slot filling directly enhance user interaction quality and task success.

Data availability

The research data supporting the findings of this study are publicly available benchmark datasets for spoken language understanding (SLU). The ATIS dataset can be accessed at: https://github.com/moore3930/SlotRefine/tree/main/data/atis and the SNIPS dataset can be accessed at: https://github.com/moore3930/SlotRefine/tree/main/data/snips Both datasets are openly available for research use under their respective licenses.

Received: 4 October 2025; Accepted: 5 December 2025
Published online: 18 December 2025

References

1. He, P. et al. *Deberta:* Decoding-enhanced bert with disentangled attention. arXiv preprint arXiv:2006.03654 (2020).
2. Istaiteh, O. et al. A transformer-based e2e slu model for improved semantic parsing. In *ICASSP 2023–2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (*IEEE,* 2023).
3. Arora, G., Jain, S. & Merugu, S. Intent detection in the age of LLMs. In *Proceedings of the 2024 conference on empirical methods in natural language processing: Industry track* (2024).
4. Kumar, N. & Baghel, B. K. Smart stacking of deep learning models for granular joint intent-slot extraction for multi-intent SLU. *IEEE Access* **9**, 97582–97590 (2021).
5. Zailan, A. S. M. et al. State of the art in intent detection and slot filling for question answering system: A systematic literature review. *Int. J. Adv. Comput. Sci. Appl.* **14**(11), 15–27 (2023).
6. Weld, H. et al. A survey of joint intent detection and slot filling models in natural language understanding. *ACM Comput. Surv.* **55**(8), 1–38 (2022).

7. Goo, C.-W. et al. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: Human language technologies, Volume 2 (Short Papers)* (2018).
8. Han, S.C. et al. Bi-directional joint neural networks for intent classification and slot filling. In *the Proceedings of the Conference of the International Speech Communication Association (INTERSPEECH'21)* ISCA (2022).
9. Wu, D. et al. CEA-Net: A co-interactive external attention network for joint intent detection and slot filling. *Neural Comput. Appl.* 1–13 (2024).
10. Zhang, X. et al. High-similarity sheep face recognition method based on a Siamese network with fewer training samples. *Comput. Electron. Agric.* **225**, 109295 (2024).
11. Chinnasamy, K. et al. DKSCNN: Deep Kronecker Siamese convolutional neural network enabled speaker identification. *Expert Syst. Appl.* **288**, 127946 (2025).
12. Huang, H., Feng, X. & Wan, Z. Joint model of intent recognition and slot filling based on graph neural network fusion of external knowledge base. In *2024 36th Chinese Control and Decision Conference (CCDC)* (IEEE, 2024).
13. Mao, J., Hang, J. -Y. & Zhang, M. -L. Learning label-specific multiple local metrics for multi-label classification. In *IJCAI* (2024).
14. Agarwal, V. et al. *Lidsnet:* A lightweight on-device intent detection model using deep siamese network. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)* (IEEE, 2021).
15. Yang, S. et al. Few-shot intent detection with mutual information and contrastive learning. *Appl. Soft Comput.* **167**, 112338 (2024).
16. Chen, et al. Robust adaptive feature enhancement and contrastive clustering for new intent discovery. In *2024 International Joint Conference on Neural Networks (IJCNN)* (IEEE, 2024).
17. Zhang, X. et al. A contrastive learning-based task adaptation model for few-shot intent recognition. *Inf. Process. Manage* **59**(3), 102863 (2022).
18. Xu, J. et al. Dual contrastive learning for cross-domain named entity recognition. *ACM Trans. Inf. Syst.* **42**(6), 1–33 (2024).
19. Soleymanbaigi, S. et al. Encoder-Decoder nonnegative matrix factorization with β-divergence for data clustering. *Pattern Recogn.* **171**, 112211 (2025).
20. Rafiepour, M. & Sartakhti, J. S. CTRAN: CNN-transformer-based network for natural language understanding. *Eng. Appl. Artif. Intell.* **126**, 107013 (2023).
21. Mikolov, T., Yih, W. -t. & Zweig, G. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: Human language technologies* (2013).
22. Ren, F. & Xue, S. Intention detection based on siamese neural network with triplet loss. *IEEE Access* **8**, 82242–82254 (2020).
23. Wang, Y., Tang, L. & He, T. Attention-based CNN-BLSTM networks for joint intent detection and slot filling. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data: 17th China National Conference, CCL 2018, and 6th International Symposium, NLP-NABD 2018, Changsha, China, October 19–21, 2018, Proceedings 17* (Springer, 2018).
24. Karim, A. R. & Uzuner, O. Leveraging machine-generated data for joint intent detection and slot filling in bangla: A resource-efficient approach. In *Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL 2025)* (2025).
25. Pham, T. & Nguyen, D. Q. JPIS: A Joint Model for Profile-Based Intent Detection and Slot Filling with Slot-to-Intent Attention. In *ICASSP 2024–2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2024).
26. Luo, B. & Feng, B. Bi-directional joint model for intent detection and slot filling. In *2024 7th International Conference on Advanced Algorithms and Control Engineering (ICAACE)* (IEEE, 2024).

Acknowledgements

The authors extend their heartfelt gratitude to PETRONAS for funding the research under UTP Foundation—Fundamental Research Grant (015LC0-524)

Author contributions

Y.I.M: Conceive the idea, and wrote the main manuscript N.S: Supervise the work A.Z: Co-supervise the work M.N: Fund acquisition A.S.H: Fund acquisition S.H.H: Co-supervise work Y.A.B: Fund acquisition.

Funding

This research is funded by PETRONAS under UTP Foundation—Fundamental Research Grant (015LC0-524).

Declarations

Competing interests

The authors declare that they have no competing interests.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

During the preparation of this work, the authors used ChatGPT to assist with grammar correction, language refinement, and clarity improvements. After using this tool, the authors carefully reviewed and edited the content as needed and take full responsibility for the final version of the manuscript.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-025-31864-8.

Correspondence and requests for materials should be addressed to Y.I.M.