applied sciences

MDPI

*Article*

# Computational Linguistics with Deep-Learning-Based Intent Detection for Natural Language Understanding

**Hala J. Alshahrani [1], Khaled Tarmissi [2], Hussain Alshahrani [3], Mohamed Ahmed Elfaki [3], Ayman Yafoz [4], Raed Alsini [4], Omar Alghushairy [5] and Manar Ahmed Hamza [6,*]**

1 Department of Applied Linguistics, College of Languages, Princess Nourah Bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
2 Department of Computer Sciences, College of Computing and Information System, Umm Al-Qura University, Mecca 24382, Saudi Arabia
3 Department of Computer Science, College of Computing and Information Technology, Shaqra University, Shaqra 11961, Saudi Arabia
4 Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 22254, Saudi Arabia
5 Department of Information Systems and Technology, College of Computer Science and Engineering, University of Jeddah, Jeddah 21589, Saudi Arabia
6 Department of Computer and Self Development, Preparatory Year Deanship, Prince Sattam Bin Abdulaziz University, Al-Kharj 16278, Saudi Arabia
* Correspondence: ma.hamza@psau.edu.sa

**Abstract:** Computational linguistics explores how human language is interpreted automatically and then processed. Research in this area takes the logical and mathematical features of natural language and advances methods and statistical procedures for automated language processing. Slot filling and intent detection are significant modules in task-based dialogue systems. Intent detection is a critical task in any natural language understanding (NLU) system and constitutes the base of a task-based dialogue system. In order to build high-quality, real-time conversational solutions for edge gadgets, there is a demand for deploying intent-detection methods on devices. This mandates an accurate, lightweight, and fast method that effectively operates in a resource-limited environment. Earlier works have explored the usage of several machine-learning (ML) techniques for detecting intent in user queries. In this article, we propose Computational Linguistics with Deep-Learning-Based Intent Detection and Classification (CL-DLBIDC) for natural language understanding. The presented CL-DLBIDC technique receives word embedding as input and learned meaningful features to determine the probable intention of the user query. In addition, the presented CL-DLBIDC technique uses the GloVe approach. In addition, the CL-DLBIDC technique makes use of the deep learning modified neural network (DLMNN) model for intent detection and classification. For the hyperparameter tuning process, the mayfly optimization (MFO) algorithm was used in this study. The experimental analysis of the CL-DLBIDC method took place under a set of simulations, and the results were scrutinized for distinct aspects. The simulation outcomes demonstrate the significant performance of the CL-DLBIDC algorithm over other DL models.

**Keywords:** computational linguistics; deep learning; natural language understanding; intent detection; mayfly optimization



**Citation:** Alshahrani, H.J.; Tarmissi, K.; Alshahrani, H.; Ahmed Elfaki, M.; Yafoz, A.; Alsini, R.; Alghushairy, O.; Ahmed Hamza, M. Computational Linguistics with Deep-Learning-Based Intent Detection for Natural Language Understanding. *Appl. Sci.* **2022**, *12*, 8633. https://doi.org/10.3390/app12178633

Academic Editors: Valentino Santucci and Paolo Mengoni

Received: 5 August 2022
Accepted: 25 August 2022
Published: 29 August 2022



## 1. Introduction

With the development of the task-based dialogue mechanism, natural language understanding (NLU), as a critical element of the task-based dialogue system, has gained more interest among researchers [1]. An author could capture context data for identifying the intent of a user by utilizing intellectual, interactive gadgets which talk to humans, in various cases, and then derive the semantic constituents from the text that the end-user

inputs into semantic slots which have been previously defined [2]. Two modules, such as slot filling and intent detection, could transform the text into a semantic representation that offers the task data to support the dialogue system and aids users in achieving their goals [3]. "Intent detection" refers to the task of categorizing natural language utterances into semantic intent classes that have previously been defined [4].

Finding message intent in natural language utterances becomes a critical task for conversational mechanisms [5]. In applications ranging from natural-language response production to offering intellectual proposals, understanding the primary intent of a context interaction becomes critical. Conventionally, research on intent detection has focused on this task with the supposition that inference and training are executed using cloud infrastructure or well-equipped servers [6,7]. This results in the inappropriateness of prevailing machine-learning (ML) techniques for real-time dialogue mechanisms in low-resource edge devices because of its higher latency and dependence on large, pre-trained methods. Recently, there has been greater commercial and academic interest in guiding artificial intelligence (AI) solutions that could work in a straightforward manner on a device using local data [8,9]. On-device AI methods are capable of supporting intent detection in real-time at a low latency and aid in guarding the privacy of delicate user data, such as smartphone messages.

Slot filling and intent detection have always been implemented as the pipeline modules in research on conventional, task-based dialogue systems [10]. However, pipeline techniques propagate mistakes effortlessly. In contrast with pipeline techniques, joint methods have the benefit of using the dependency between slots and intent. Such works have been primarily classified into two classes: the use of semantic analysis and a joint model that incorporates both classification and feature design into the learning process [11]. However, few studies have used additional information, such as the incidence relation and the shared resources between slot-filling and intent-detection modules. However, the above-mentioned studies contain a typical issue: that is, the high-quality and standardized public datasets are highly inadequate because labeling and collecting datasets require more effort and time [12]. Thus, researchers have tried to include an external knowledge base in the prevailing datasets so as to solve the issue of data scarcity. A knowledge base can be engineered into a knowledge cluster, and thus it becomes structured, easy to use and operate, and provides comprehensive and organized knowledge [13]. Moreover, it is a set of interrelated knowledge pieces that are managed, stored, used, and organized in computer memory by some knowledge-representation mode to solve issues in certain domains. Recent research studies have shown that the impact of NLU is enhanced by the presentation of a knowledge base.

This article proposes Computational Linguistics with Deep-Learning-Based Intent Detection and Classification (CL-DLBIDC) for NLU. The CL-DLBIDC technique receives word-embedding as input and uses learned meaningful features to determine the probable intention of the user query. In addition, the CL-DLBIDC technique uses the GloVe approach. In addition, the CL-DLBIDC technique makes use of the deep learning modified neural network (DLMNN) model for intent detection and classification. For the hyperparameter tuning process, the mayfly optimization (MFO) algorithm has been used in this study. The experimental analysis of the CL-DLBIDC technique took place under a set of simulations, and distinct aspects of the results have been examined.

## 2. Related Works

Yan et al. [14] have presented a semantic-enhanced Gaussian mixture method (SEG) for unknown intent identification. In particular, the authors devised utterance embeddings that have a Gaussian mixture distribution, and they infused dynamic-class semantic data into Gaussian means that allow for the learning of more class-concentrated embeddings, which can be helpful in facilitating downstream outlier recognition. Coupled with a density-related, outlier-identification method, SEG has attained competitive outcomes on three real, task-based, dialogue datasets in two languages for unknown intent recognition. Gangad-

haraiah and Narayanaswamy [15] have examined an attention-related neural network (NN) method which executes multi-label categorization to identify many intents and generates labels for intents, as well as slot-labels, at the token-level. The authors displayed the existing performance for slot-label identification and intent detection through a comparison with strong, newly formulated methods.

Dopierre et al. [16] have presented ProtAugment, a meta-learning technique for short text classification (an intent-detection task). ProtAugment is a new extension of the Prototypical Network, which confines overfitting to the bias presented by a few-shots classifier objective at every episode. It depends on various paraphrasing. It is a conditional language method that has become the initial, fine-tuned language for paraphrasing, and diversity was later presented at the decoding phase of every meta-learning episode. Nigam et al. [17] have proposed a method for forecasting slots and the intent of queries for chatbots that will answer career-based queries. The authors considered a multistage technique in which both the process (slot-tagging and intent-classification) imitate one another's decision-making at various levels. The method segregates the issue into stages, figuring out single issues, and transferring the related outcomes of the present stage to the following stage, thus minimizing the search space for consequent levels, and finally, it performs categorizations and tagging that are more viable after every stage.

In [18], the authors began from the nature of out-of-domain (OOD) intent classifications and predicted its optimizing objective. The authors even presented an effective technique, called k-nearest-neighbor (KNN) contrastive learning. This technique uses KNN for learning discriminatory semantic features that are highly conducive to OOD identification. [19] enhanced intent-detecting performance by using datasets that were gathered in the same field, but which have various intent groups. In the baseline mechanism, the authors used the existing, pre-trained transforming techniques with an intent-identification-classifier layer. This technique contains two fine-tuned stages. In the initial stage, the variables of the classifier and transformer layers were adapted utilizing an in-domain dataset to the targeted set. Then, the author retained the transformer parameters, which were updated, but replaced the classifier with a newer one designed for the targeted task. In the second stage, the network was finely tuned endwise, utilizing the targeted set.

Shen et al. [20] modeled an innovative domain-regularized module (DRM) for reducing the overconfident phenomenon of a vanilla classifier, attaining superior generalization in both cases. In addition, DRM was utilized as a drop-in replacement for the final layer in any NN-related intent classifier, offering a low-cost method for an important enhancement. Qin et al. [21] introduced an Adaptive Graph-Interactive Framework (AGIF) for joint, multiple-intent recognition and slot-filling, where the authors introduced an intent–slot graph communication layer for modeling strong relations among the intents and slots. Such communication layers can be implemented for every token adaptively, which has the benefit of mechanically deriving the related intent data, forming a finely grained intent data compilation for token-level slot estimation.

## 3. The Proposed Model

This study presents a novel CL-DLBIDC approach to detecting and classifying intents for natural language understanding. The presented CL-DLBIDC technique receives word embedding as input and uses learned meaningful features to determine the probable intention of the user query. In addition, the CL-DLBIDC technique uses the GloVe approach. In addition, the CL-DLBIDC technique makes use of the DLMNN model for intent detection and classification. For the hyperparameter tuning process, the MFO algorithm was used in this study. Figure 1 demonstrates the overall block diagram of the CL-DLBIDC method.

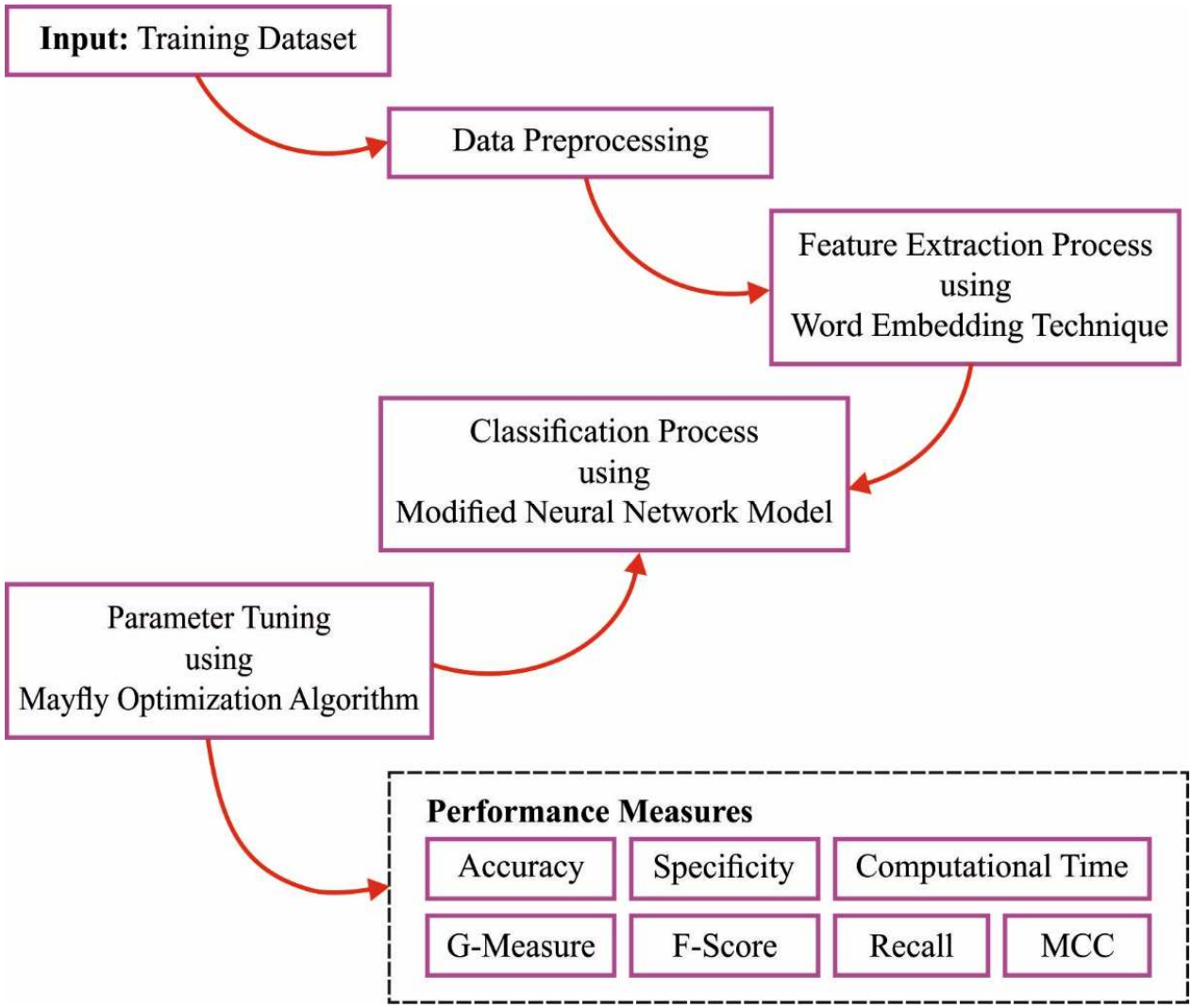


**Figure 1.** Overall block diagram of the CL-DLBIDC approach.

### *3.1. Word Embedding*

Here, the CL-DLBIDC technique uses the GloVe approach. The GloVe (global vector) model exploits global co-occurrence matrixes to generate representations of word vectors [22]. This study focuses on the co-occurrence of words in the corpora. Hence, words which have the same context would be close to one another. Because antonymous words share similar contexts and they close to one another, the output can be semantically wrong. This kind of semantic incorrectness of the vector space must be solved in order to achieve good outcomes. The first embedding is enriched by a somewhat-adapted form of an enrichment method. The resultant vector space will be semantically correct when compared to the first one. Consider that $v = \{v_1, v_2, v_n\}$ represents the group of word vectors, where $v_i$ indicates the vector for $j^{th}$ words in a vocabulary. Assume $A$, $S$, $R$, and $H$ denote the collection of indices for pairs of antonymous, synonymous, related, and hyponymous or hypernymous words, correspondingly. For instance, if $(i, j)$ is a component of $S$, later $i^{th}$ and $j^{th}$ words in the vocabulary are synonyms. The first vector space $y$ is adapted through objective function and the set (constraint) to attain a novel vector space $V'$, where $V' = \{v'_1, v'_2, v'_n\}$. The formula of objective functions is given in the following section.

3.1.1. Enrichment Using Synonyms

Equation (1) brings synonymous words close to one another in the vector space by improving the cosine similarities among them.

$$SE(V') = \sum_{(i,j)\varepsilon S} \tau(\delta - \cos(v_i, v_j)) \quad (1)$$

$S$ represents the collection of pairs of synonyms in a vocabulary, and it involves synonyms from WordNet and equivalence relations. $\delta$ is fixed to one and $\tau(x) = \max(O, x)$.

3.1.2. Enrichment Using Antonyms

Equation (2) moves antonymous words farther from one another by decreasing the cosine similarities among them to 0 or less.

$$AE(V') = \sum_{(i,j)\varepsilon A} \tau(\cos(v_i, v_j)) \quad (2)$$

$A$ represents the collection of pairs of antonyms that involve an exclusion relation from PPDB and antonyms from WordNet.

3.1.3. Enriching with Related Words

Related words have a similar sense; however, they are not synonyms in a lexicon (e.g., "ride" vs. "commute"). Bringing related words close to one another in a vector space would be useful because they have similar senses. The objective function shown in Equation (3) raises cosine similarities among related words to 0 or greater than 0.

$$RE(V') = \sum_{(i,j)\varepsilon R} \tau(\delta - \cos(v_i, v_j)) \quad (3)$$

$R$ denotes the group of related words from *The Macmillan Dictionary*, and $\delta$ was fixed to 0. The value of $\delta$ guarantees that similarities among related words are 0 or greater than 0. This ensures that related words are not as close to one another as synonyms are.

3.1.4. Enriching with Hypernyms and Hyponyms

Hypernymous terms have wider meanings, whereas hyponymous terms have particular meanings. A hypernym–hyponym is a pair of common terms and its particular categories (e.g., "book"–"recipe book"). The cosine similarities between these words must be 0 or greater than 0. The objective function provided in Equation (4) is the same as that is Equation (3), and it is utilized for achieving this.

$$HE(V') = \sum_{(i,j)\varepsilon H} \tau(\delta - \cos(v_i, v_j)) \quad (4)$$

$H$ is the group of hypernym–hyponym pairs in vocabulary $V$, with a value of $\delta$ as 1 and a path length of 1 in WordNet.

3.1.5. Regularization

When a word vector is adapted by employing the above for constraint, it becomes distinct from the prior, neighboring vector. It alters the semantics of embedding, and regularization must be performed to retain the meaning. The converted vector space must be bent to the original vector space to conserve the meaning. The objective function

provided in Equation (5) keeps cosine similarities among its neighbors and an initial word vector greater than or equivalent to the cosine similarities among them.

$$VS\ P(V,\ V') = \sum_{(i,j)\varepsilon A}^{M} \sum_{j\varepsilon N(i)} \tau\left(\cos\left(v_i',\ v_j'\right) - \cos\left(v_i,\ v_j\right)\right) \tag{5}$$

$M$ denotes the word count in vocabulary, and $N(i)$ denotes the group of neighbors of $i^{th}$ in the first embedding space. $N(i)$ involves the word in $V$ whose cosine similarities among $i^{th}$ words is higher than 0.8.

The last objective function is a linear integration of the abovementioned terms, as shown in Equation (6).

$$E(V,\ V') = SE(V') + AE(V') + RE(V') + HE(V') + VSP(V,\ V') \tag{6}$$

The objective function is minimalized by the stochastic gradient descent (SGD) approach for transformed vector $V'$ and initial vector $V$. SGD ran for 20 epochs with a learning rate of 0.1.

### 3.2. Intent Detection Using the DLMNN Model

At this stage, the CL-DLBIDC technique makes use of the DLMNN model for intent detection and classification. In the DLMNN model, all of the inputs are offered to separate nodes that are common from the input region of classification [23]. The weight signifies a randomly allocated value and can be connected with all of the inputs. The following is the Hidden Layer (HL). The nodes in this HL are termed "hidden nodes", and they apply the function of summing the product of input values and the weighted vector of every input node which is connected to it.

In DLMNN, the weighted value is optimized by employing the MFO algorithm and DLMNN. An arbitrary weighted value offers a further backpropagation (BP) procedure for achieving the outcome in an optimized manner. The activation function is then executed, and the resultant layer is carried to following layer. This weight is an utmost control on the classification result. The algorithmic steps of DLMNN classification are as follow:

Step 1: Allocate score value of selective features and its corresponding weight as:

$$E_i = \{E_1, E_2, E_3 \ldots E_n\}, \tag{7}$$

$$W_i = \{W_1,\ W_2,\ W_3 \ldots W_n\}, \tag{8}$$

Step 2: Multiply the input with the weighted vectors, which are arbitrarily chosen, and then, it can be entirely summed as:

$$R = \sum_{i=1}^{n} E_i W_i, \tag{9}$$

where $R$ refers to the summed value, $E_i$ denotes the input entropy value, and $W_i$ stands for the weight value.

Step 3: Estimate the activation function (AF), employing

$$AF_i = f\left(\sum_{i=1}^{n} E_i W_i\right), \tag{10}$$

Step 4: Measure the resultant HL as:

$$Y_i = A_i + \sum G_i W_i, \tag{11}$$

where $A_i$ defines the bias value, $W_i$ demonstrates the weight between the input and HLs, and $G_i$ refers to the values that are changed by application of $AF$.

Step 5: Re-perform the earlier three steps on all of the layers of the DLMNN. Last, estimate the resultant unit by adding up every input signal weight to attain the resultant layer neuron value,

$$Ou_i = A_i + \sum P_i W_i, \tag{12}$$

where $P_i$ signifies the value of the layer which leads to the output 1, $W_j$ represents the weight of HL, and $Ou_i$ demonstrates the output unit.

Step 6: Contrast the network outcome with the target values. The variance among these two values produces the error signal. This value can be demonstrated mathematically as:

$$E_r = Ta_i - Ou_i, \tag{13}$$

where, $E_r$ represents the error signal, $Ta_i$ indicates the aimed target output, and $Ou_i$ stands for the classification current output.

Step 7: At this point, the resultant unit is weighted against the target values. Accordingly, the relative error is defined. According to this error, a value $\delta_i$ has been calculated, and it can be utilized for allocating the error at the resultant layer back to every other unit on network:

$$\delta_i = E_r[f(Ou_i)], \tag{14}$$

Step 8: The weighted correction is measured by employing the BP approach. This relation is:

$$wc_i = \beta\delta_i(E_i), \tag{15}$$

where $wc_i$ defines the weighted correction, $\beta$ signifies the momentum term, $E_i$ implies the input vector, and $\delta_i$ stands for the error that is distributed from the network.

### *3.3. Parameter Tuning Using the MFO Algorithm*

For the hyperparameter tuning process, the MFO algorithm was used in this study. The major idea of this method is based on the flight and mating behaviors of the mayfly (MF) [24]. It integrates the benefits of the firefly algorithm (FA), particle swarm optimization (PSO), and the genetic algorithm (GA). Initially, the MFO algorithm generates an MF population encompassing males and females. The existing velocity and location of *i-th* MFs are two $n$-dimensional vectors; they are correspondingly represented as $v_i = (v_{i1}, v_{i2} \ldots, v_{in})$ and $x_i = (x_{i1}, x_{i2} \ldots, x_{in})$. Every MF adapts its location based on its individual best $(pbest)$ location and the optimal $(gbest)$ location originated by the entire MF swarm.

Male MFs cluster together, which recommends that the position is upgraded on the basis of social and personal experience. $x_i^t$ denotes the current location of *i-th* male MFs at $t$ iteration. Assume that $x_i^t$ denotes the existing location of *i-th* MFs at $t$ iteration; the location is upgraded by adding velocity $v_i^t$, and it is shown below:

$$x_i^{t+1} = x_i^t + v_i^t. \tag{16}$$

Assume that male MFs often implement their mating dance not distant from the water, and it is updated in the following equation:

$$v_{ij}^{t+1} = gv_{ij}^t + a_1 e^{-\beta r_p^2}\left(pbest_{ij} - x_{ij}^t\right) + a_2 e^{-\beta r_g^2}\left(gbest_j - x_{ij}^t\right), \tag{17}$$

In Equation (17), $\beta$ refers to the visibility coefficient, $a_1$ and $a_2$, determined by the positive constant that represents the attraction. $pbest_{ij}$ indicates the optimal location attained using the *i-th* male MFs in $j$ dimension. $r_p$ and $r_g$ indicate the Euclidean distance between $\chi_j$ and $pbest_j$, and between $\chi_j$ and $gbest$, respectively. The gravity coefficient is

denoted as $g$, which is a fixed number among [0, 1], or it is steadily lowered over the iteration, as follows:

$$g = g_{\max} - \frac{g_{\max} - g_{\min}}{iter_{\max}} \times iter, \tag{18}$$

In Equation (18), $g_{\max}$ and $g_{\min}$ correspondingly embody the minimum and maximum weights. The current and the overall number of iterations are represented as *iter* and $ter_{\max /}$. The personal optimal location $pbest_i$ at iteration $t+1$ is defined as follows:

$$pbest_i = \begin{cases} x_i^{t+1}, iff\left(x_i^{t+1}\right) < f(pbest_i) \\ same\ as\ before,\ otherwise. \end{cases} \tag{19}$$

The finest male MF continues implementing up and down movements at distinct velocities, and this equation is shown below:

$$v_{ij}^{t+1} = v_{ij}^t + d \times r, \tag{20}$$

In Equation (20), the mating dance coefficient can be represented as $d$, and $r$ denotes an arbitrary number within $[-1, 1]$. Female MFs do not cluster together, but they move to male MFs. $y_i^t$ denotes the current location of the *i-th* female MF at $t$ iteration. The change of *i-th* female MF in location is evaluated by Equation (21):

$$y_i^{t+1} = y_i^t + v_i^t. \tag{21}$$

In the MFO algorithm, female and male MFs with similar individual fitness rankings attract one another, and the female MF's location changes in terms of the position of the male MF with a similar ranking, as follows:

$$v_{ij}^{t+1} = \begin{cases} gv_{ij}^t + a_2 e^{-\beta r_{mf}^2}\left(x_{ij}^t - y_{ij}^t\right),\ if\ f(y_i) > f(X_i) \\ gv_{ij}^t + fl \times r,\ if\ f(y_i) \le f(xi)\ , \end{cases} \tag{22}$$

where $v_{ij}^t$ and $y_{ij}^t$ denote the *i-th* female MF's velocity and location in $j$ dimension at $t$ iteration. $a_2$ and $\beta$ indicate the attraction constant and visibility coefficient, correspondingly. $r_{mf}$ refers to the Euclidean distance between the *i-th* female MF and *i-th* male MF, $fl$ denotes a random walking coefficient that recommends that a female is not attracted to a male, and $r$ implies an arbitrary number within $[-1, 1]$. Figure 2 illustrates the steps involved in the MFO technique.

The act of mating can be denoted as the crossover operator that offers the global search ability for the MFO algorithm. In the MFO algorithm, *female* and *male* MFs with similar individual fitness rankings mate with one another to generate offspring. The mating process of every pair of female and male MFs generates two offspring, and the crossover operator is given below:

$$\begin{cases} \gamma_1 = L \times male + (1 - L) \times female \\ \gamma_2 = L \times female + (1 - L) \times male, \end{cases} \tag{23}$$

In Equation (23), $L$ implies an arbitrary number within [0, 1]. Furthermore, *female* and *male* denote the parents. First, the offspring velocity is fixed as 0.

The mutation operator gives partial local search ability to MFO algorithm. The selected offspring is upgraded by adding an arbitrary number to the parameter over this method; hence, the offspring is changed to

$$\gamma_n' = \gamma_n + \sigma N_n(0, 1), \tag{24}$$

In Equation (24), the standard deviation and uniform distribution are signified as $\sigma$ and $N_n$, correspondingly. Even though this operator provides the MFO algorithm with

robust local search ability, $d$ and $fl$ should be slowly decreased. Both values are upgraded over the $t$ iterations as follows:

$$d_t = d_0 \delta^t \tag{25}$$

$$fl_t = fl_0 \delta^t, \tag{26}$$

Here, $\delta$ is a fixed value among [0, 1].

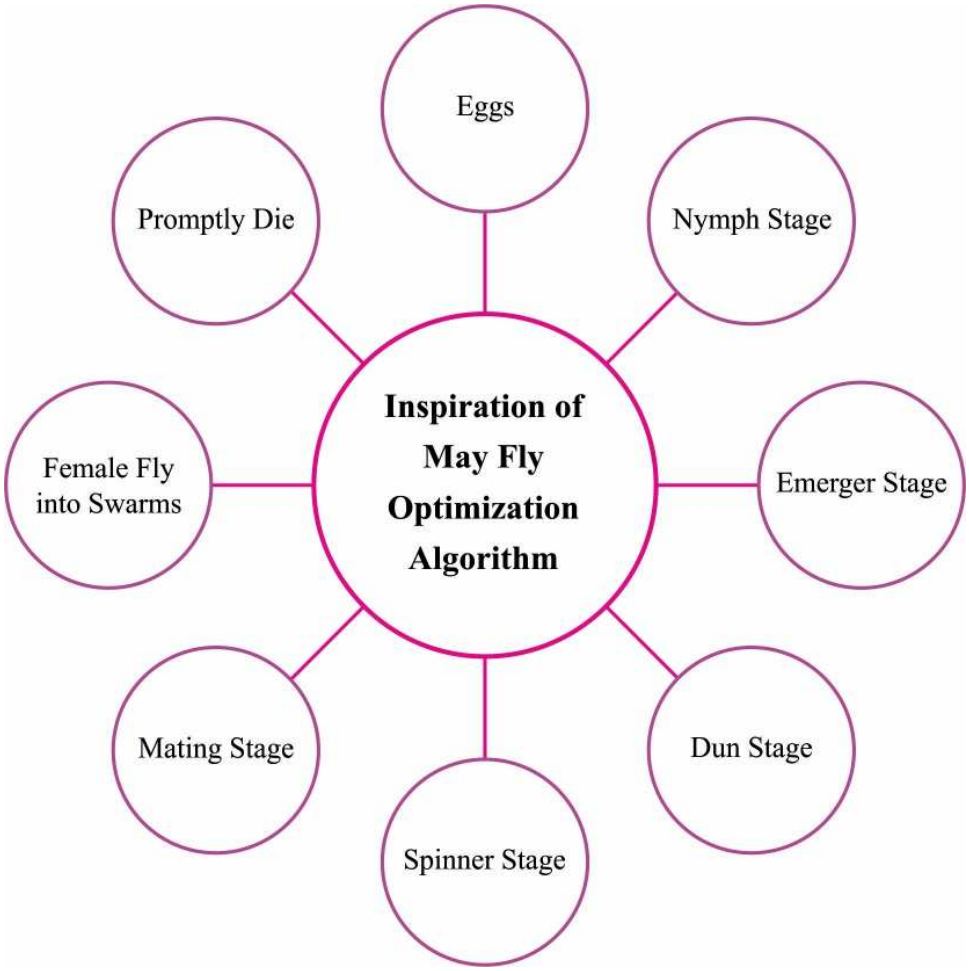


**Figure 2.** Steps involved in MFO.

## 4. Experimental Evaluation

The intention detection results of the CL-DLBIDC model were tested using the SNIPS dataset [25]. In this work, we considered 3500 samples with 7 class labels, as depicted in Table 1.

**Table 1.** Dataset details.

| Label | Class | No. of Samples |
|---|---|---|
| 1 | CreativeWork | 500 |
| 2 | PlayMusic | 500 |
| 3 | RateBook | 500 |
| 4 | ScreeningEvent | 500 |
| 5 | GetWeather | 500 |
| 6 | AddToPlaylist | 500 |
| 7 | BookRestaurant | 500 |
| **Total No. of Samples** | | **3500** |

Figure 3 depicts the classification outcomes of the CL-DLBIDC model in terms of the confusion matrix under distinct sizes of training (TR) data and testing (TS) data. The CL-DLBIDC model classified all of the distinct class labels of the intents under all sizes of TR and TS data.

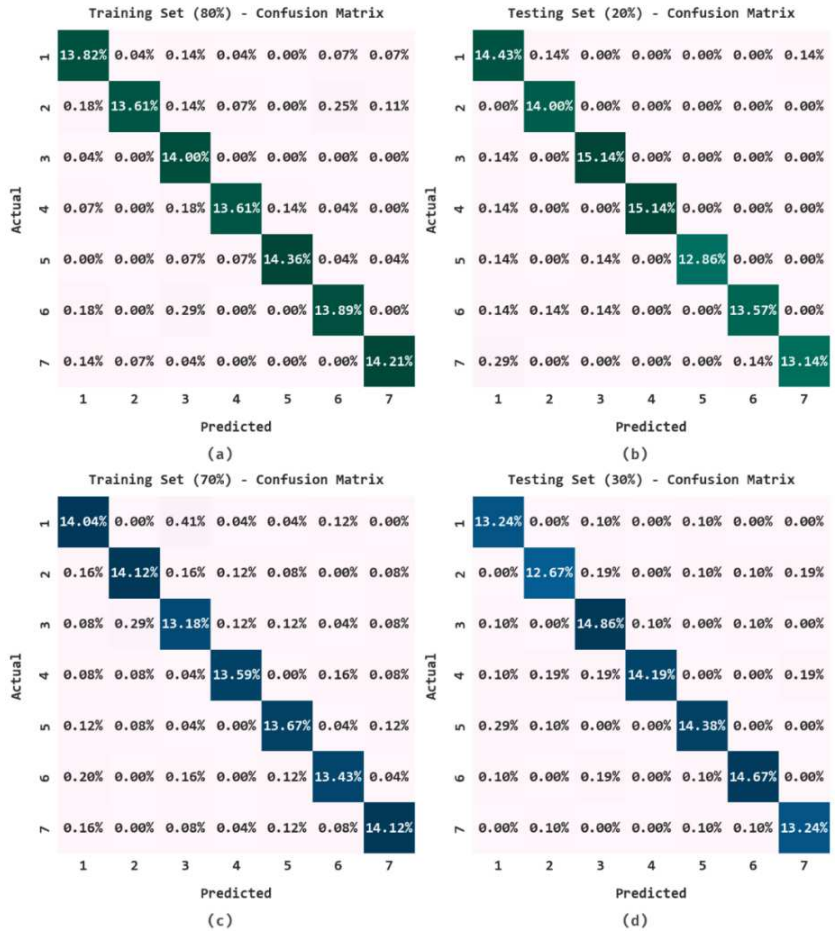


**Figure 3.** Confusion matrices of CL-DLBIDC: (**a**) 80% of TR data, (**b**) 20% of TS data, (**c**) 70% of TR data, and (**d**) 30% of TS data.

Table 2 exhibits a brief set of data regarding the intent-detection performance of the CL-DLBIDC model on 80% of the TR data and 20% of the TS data. These results depict that the CL-DLBIDC model has attained enhanced values under all classes. For instance, with Class 1, the CL-DLBIDC method has granted $accu_y$, $reca_l$, $spec_y$, $F_{score}$, the Matthews Correlation Coefficient (MCC), and $G_{measure}$ of 99.04%, 97.48%, 99.29%, 96.63%, 96.07%, and 96.63%, respectively.

Table 3 displays a detailed set of data on the intent-detection performance of the CL-DLBIDC method on 70% of the TR data and 30% of the TS data. The experimental values indicate that the proposed model has shown enhanced performance on the classification of distinct classes on 70:30% of the TR/TS data.

**Table 2.** Result analysis of the CL-DLBIDC approach with distinct measures under 80:20 of TR/TS data.

| Labels | Accuracy | Recall | Specificity | F-Score | MCC | G-Measure |
|---|---|---|---|---|---|---|
| **Training Set (80%)** | | | | | | |
| 1 | 99.04 | 97.48 | 99.29 | 96.63 | 96.07 | 96.63 |
| 2 | 99.14 | 94.78 | 99.87 | 96.95 | 96.48 | 96.97 |
| 3 | 99.11 | 99.75 | 99.00 | 96.91 | 96.44 | 96.95 |
| 4 | 99.39 | 96.95 | 99.79 | 97.82 | 97.47 | 97.82 |
| 5 | 99.64 | 98.53 | 99.83 | 98.77 | 98.56 | 98.77 |
| 6 | 99.14 | 96.77 | 99.54 | 97.01 | 96.51 | 97.01 |
| 7 | 99.54 | 98.27 | 99.75 | 98.39 | 98.12 | 98.39 |
| **Average** | **99.29** | **97.50** | **99.58** | **97.50** | **97.09** | **97.51** |
| **Testing Set (20%)** | | | | | | |
| 1 | 98.86 | 98.06 | 98.99 | 96.19 | 95.54 | 96.21 |
| 2 | 99.71 | 100.00 | 99.67 | 98.99 | 98.83 | 98.99 |
| 3 | 99.57 | 99.07 | 99.66 | 98.60 | 98.35 | 98.61 |
| 4 | 99.86 | 99.07 | 100.00 | 99.53 | 99.45 | 99.53 |
| 5 | 99.71 | 97.83 | 100.00 | 98.90 | 98.74 | 98.91 |
| 6 | 99.43 | 96.94 | 99.83 | 97.94 | 97.61 | 97.94 |
| 7 | 99.43 | 96.84 | 99.83 | 97.87 | 97.55 | 97.88 |
| **Average** | **99.51** | **98.26** | **99.71** | **98.29** | **98.01** | **98.30** |

**Table 3.** Result analysis of CL-DLBIDC approach with distinct measures under 70:30 of the TR/TS data.

| Labels | Accuracy | Recall | Specificity | F-Score | MCC | G-Measure |
|---|---|---|---|---|---|---|
| **Training Set (70%)** | | | | | | |
| 1 | 98.57 | 95.82 | 99.04 | 95.16 | 94.32 | 95.16 |
| 2 | 98.94 | 95.84 | 99.47 | 96.38 | 95.76 | 96.38 |
| 3 | 98.37 | 94.72 | 98.96 | 94.17 | 93.22 | 94.17 |
| 4 | 99.22 | 96.80 | 99.62 | 97.23 | 96.78 | 97.23 |
| 5 | 99.10 | 97.10 | 99.43 | 96.82 | 96.30 | 96.82 |
| 6 | 99.02 | 96.20 | 99.48 | 96.48 | 95.91 | 96.48 |
| 7 | 99.10 | 96.65 | 99.52 | 96.92 | 96.39 | 96.92 |
| **Average** | **98.90** | **96.16** | **99.36** | **96.16** | **95.53** | **96.17** |
| **Testing Set (30%)** | | | | | | |
| 1 | 99.24 | 98.58 | 99.34 | 97.20 | 96.77 | 97.21 |
| 2 | 99.05 | 95.68 | 99.56 | 96.38 | 95.83 | 96.38 |
| 3 | 99.05 | 98.11 | 99.21 | 96.89 | 96.34 | 96.90 |
| 4 | 99.24 | 95.51 | 99.89 | 97.39 | 96.97 | 97.40 |
| 5 | 99.24 | 97.42 | 99.55 | 97.42 | 96.97 | 97.42 |
| 6 | 99.33 | 97.47 | 99.66 | 97.78 | 97.39 | 97.78 |
| 7 | 99.33 | 97.89 | 99.56 | 97.54 | 97.16 | 97.54 |
| **Average** | **99.21** | **97.24** | **99.54** | **97.23** | **96.78** | **97.23** |

The training accuracy (TRA) and validation accuracy (VLA) obtained by the CL-DLBIDC methodology on the test dataset are shown in Figure 4. The experimental outcome denotes that the CL-DLBIDC approach has attained maximal values of TRA and VLA. Seemingly, the VLA is greater than the TRA.

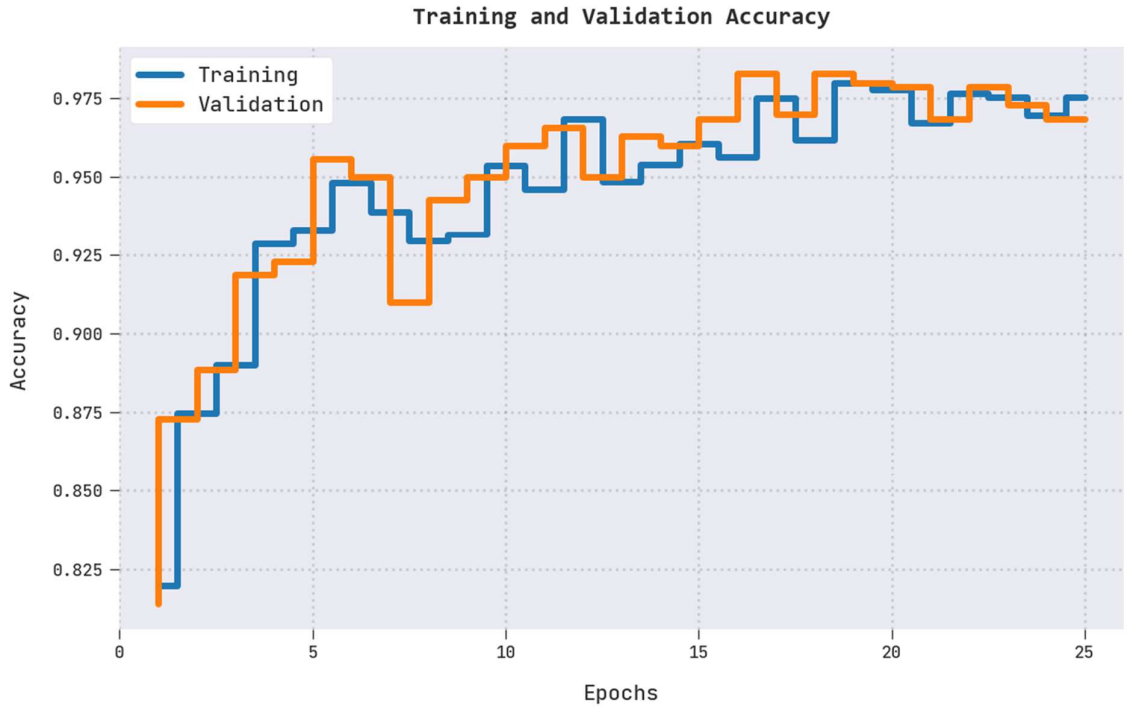


**Figure 4.** TRA and VLA analysis of the CL-DLBIDC approach.

The training loss (TRL) and validation loss (VLL) obtained by the CL-DLBIDC method on the test dataset are exhibited in Figure 5. The experimental outcome represented that the CL-DLBIDC algorithm has established minimal values of TRL and VLL. Particularly, the VLL is lesser than TRL.


**Figure 5.** TRL and VLL analysis of the CL-DLBIDC approach.

A clear precision-recall inspection of the CL-DLBIDC approach on the test dataset is portrayed in Figure 6. The figure shows that the CL-DLBIDC technique has resulted in enhanced values of precision-recall values under all classes.

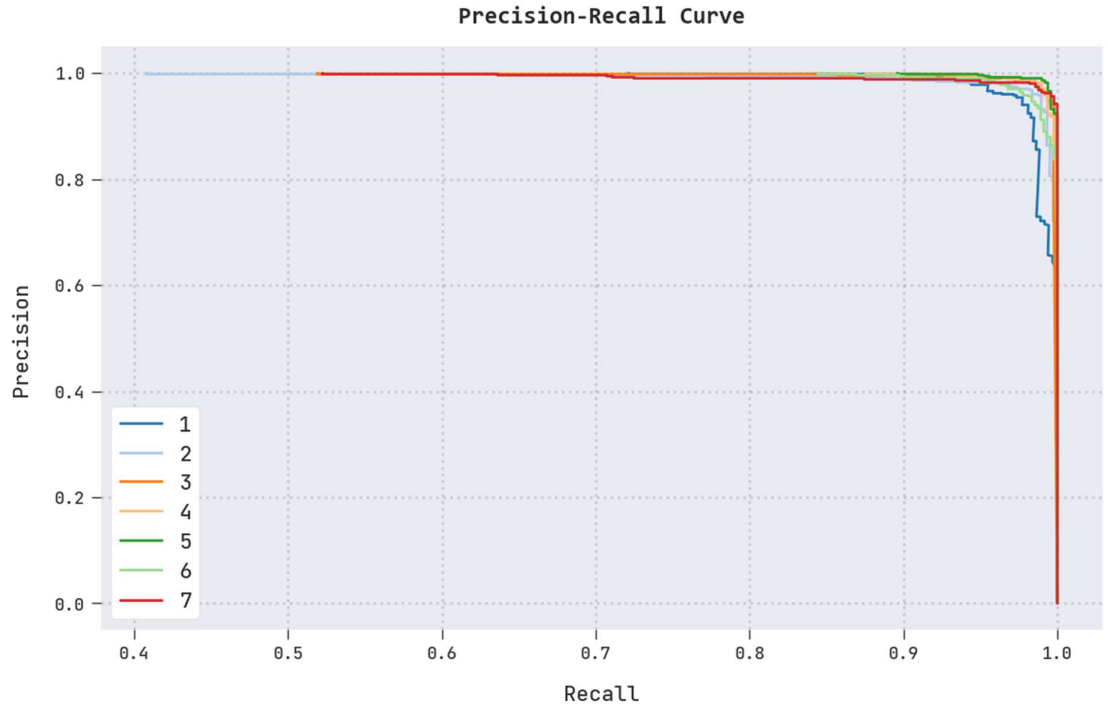


**Figure 6.** Precision-recall analysis of the CL-DLBIDC approach.

A brief ROC analysis of the CL-DLBIDC algorithm on the test dataset is presented in Figure 7. The results indicate that the CL-DLBIDC method has shown an ability to categorize distinct classes in the test dataset.

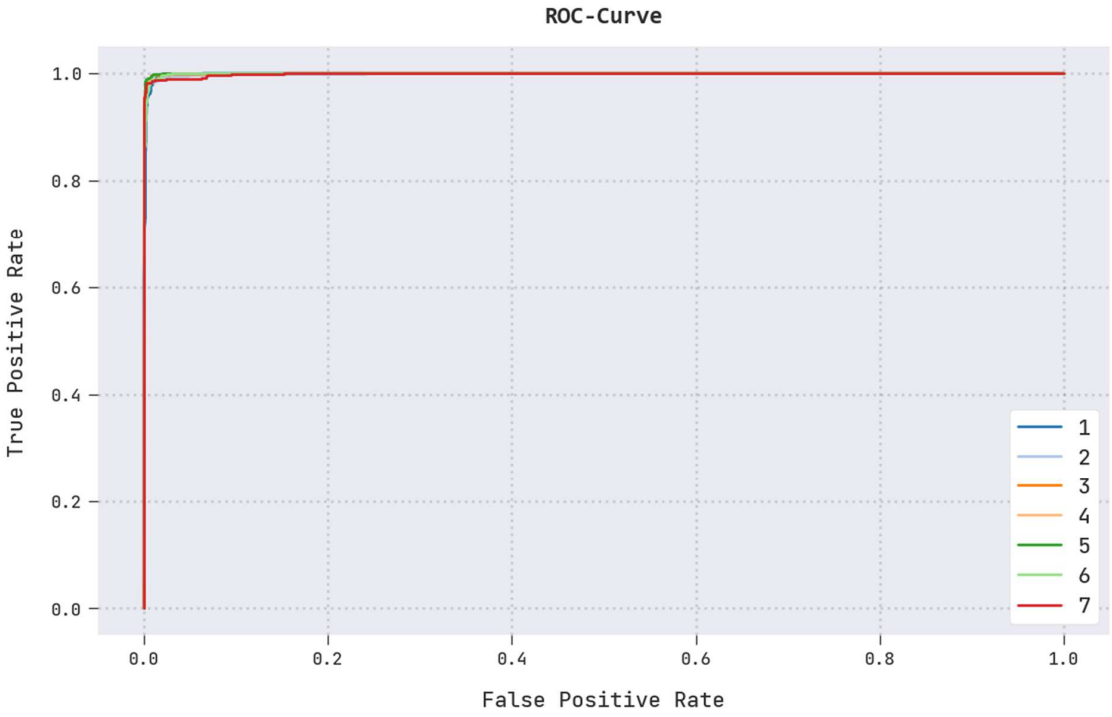


**Figure 7.** ROC analysis of the CL-DLBIDC approach.

Table 4 provides the overall classification outcome of the CL-DLBIDC model as compared with recent models, such as Lightweight on-device Intent Detection (LIDSNet), Stack-Propagation–Bidirectional Encoder Representations from Transformers (Stack-PBERT), stack-propagation, Capsule-NLU, slot-gated bidirectional long short-term memory with attention (Attention-SG-BiLSTM), and slot filling and intent detection using bidirectional long short-term memory (SFID-BLSTM) models [26]. Figure 8 showcases the comparative $accu_y$ examination of the CL-DLBIDC model with existing techniques. The figure indicates that the Attention-SG-BiLSTM model attained a lower $accu_y$ of 96.87%, whereas the LIDSNet, Capsule-NLU, and SFID-BLSTM models obtained a clearly raised $accu_y$ of 97.79%, 97.43%, and 97.36%, respectively. Next, the stack-propagation model resulted in a moderately improved $accu_y$ of 98.29%. Although the stack-PBERT model obtained a reasonable $accu_y$ of 99.25%, the CL-DLBIDC model obtained a higher $accu_y$ of 99.51%.

**Table 4.** Comparative analysis of the CL-DLBIDC approach with recent algorithms.

| Methods | Accuracy (%) | F1-Score (%) |
|---|---|---|
| CL-DLBIDC | 99.51 | 98.29 |
| Stack-PBERT | 99.25 | 97.98 |
| LIDSNet | 97.79 | 95.61 |
| Stack-Propagation | 98.29 | 96.23 |
| Capsule-NLU | 97.43 | 96.40 |
| SFID-BLSTM | 97.36 | 97.21 |
| Attention-SG-BiLSTM | 96.87 | 96.99 |

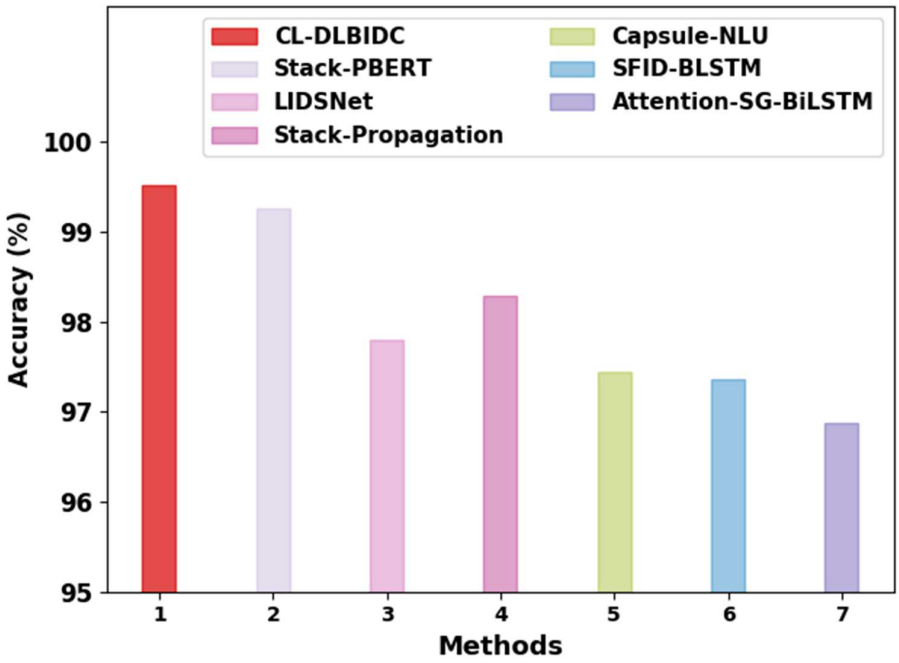


**Figure 8.** $Accu_y$ analysis of the CL-DLBIDC approach with recent algorithms.

Figure 9 showcases the comparative $F1_{score}$ analysis of the CL-DLBIDC technique with existing techniques. The figure shows that the Attention-SG-BiLSTM approach obtained a lower $F1_{score}$ of 96.99%, whereas the LIDSNet, Capsule-NLU, and SFID-BLSTM methodologies obtained a clearly raised $F1_{score}$ of 95.61%, 96.40%, and 97.21%, respectively. The stack-propagation method resulted in a moderately improved $F1_{score}$ of 96.23%. Although the stack-PBERT approach has obtained a reasonable $F1_{score}$ of 97.98%, the CL-DLBIDC methodology obtained a higher $F1_{score}$ of 98.29%. These results demonstrate the enhanced outcomes of the CL-DLBIDC model in comparison with recent models.

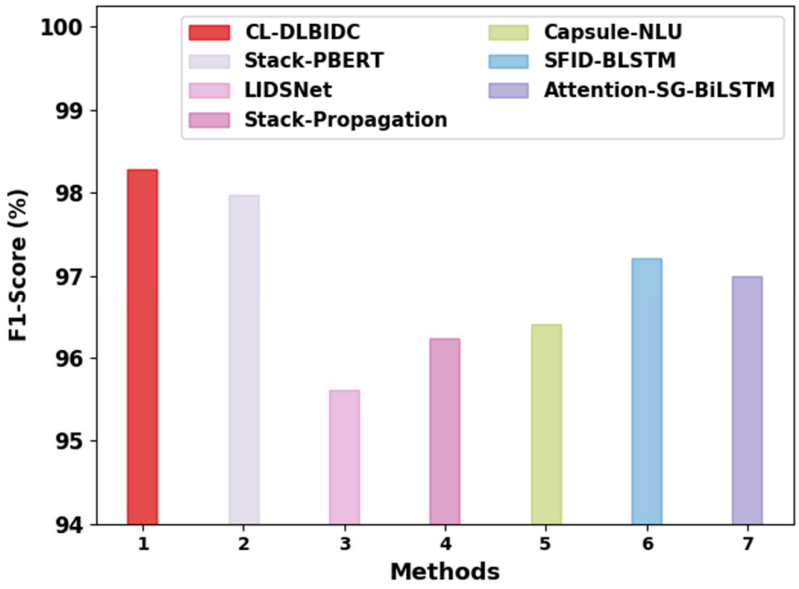


**Figure 9.** $F1_{score}$ analysis of the CL-DLBIDC approach with recent algorithms.

## 5. Conclusions

This study has presented a new CL-DLBIDC approach to detect and classify intent for natural language understanding. The CL-DLBIDC technique receives word-embedding as an input and uses learned meaningful features to determine the probable intention of the user query. In addition, the CL-DLBIDC technique uses the GloVe approach. In addition, the CL-DLBIDC technique makes use of the DLMNN model for intent detection and classification. For the hyperparameter tuning process, the MFO algorithm was used in this study. The experimental analysis of the CL-DLBIDC algorithm took place under a set of simulations, and different aspects of the results have been scrutinized. The simulation outcomes demonstrate the significant performance of the CL-DLBIDC method over other DL models, with a maximum accuracy of 99.51%. The enhanced performance of the proposed model is due to the hyperparameter tuning process using the MFO algorithm. In the future, hybrid DL models and the feature-selection process can improve the detection efficiency of the proposed model.

**Author Contributions:** Conceptualization, H.J.A.; Data curation, K.T.; Formal analysis, K.T.; Funding acquisition, M.A.H.; Investigation, H.A.; Methodology, H.J.A., H.A. and M.A.H.; Project administration, M.A.E. and M.A.H.; Resources, M.A.E. and A.Y.; Software, A.Y. and R.A.; Supervision, R.A.; Validation, O.A.; Visualization, O.A.; Writing—original draft, H.J.A.; Writing—review & editing, M.A.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R281), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work with Grant Code: (22UQU4331004DSR01).

**Institutional Review Board Statement:** This article does not contain any studies with human participants performed by any of the authors.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data sharing is not applicable to this article as no datasets were generated during the current study.

**Conflicts of Interest:** The authors declare that they have no conflict of interest. The manuscript was written with the contributions of all authors. All authors have given approval to the final version of the manuscript.

## References


1. Firdaus, M.; Golchha, H.; Ekbal, A.; Bhattacharyya, P. A deep multi-task model for dialogue act classification, intent detection and slot filling. *Cogn. Comput.* **2021**, *13*, 626–645. [CrossRef]
2. Fernández-Martínez, F.; Griol, D.; Callejas, Z.; Luna-Jiménez, C. An approach to intent detection and classification based on attentive recurrent neural networks. In Proceedings of the IberSPEECH, Valladolid, Spain, 24–25 March 2021; pp. 46–50.
3. Song, S.; Chen, X.; Wang, C.; Yu, X.; Wang, J.; He, X. A Two-Stage User Intent Detection Model on Complicated Utterances with Multi-task Learning. In Proceedings of the WebConf 2022, Lyon, France, 25–29 April 2022.
4. Al-Wesabi, F.N. Proposing high-smart approach for content authentication and tampering detection of arabic text transmitted via internet. *IEICE Trans. Inf. Syst.* **2020**, *E103*, 2104–2112. [CrossRef]
5. Weld, H.; Huang, X.; Long, S.; Poon, J.; Han, S.C. A survey of joint intent detection and slot-filling models in natural language understanding. *arXiv* **2021**, arXiv:2101.08091. [CrossRef]
6. Al-Wesabi, F.N. A hybrid intelligent approach for content authentication and tampering detection of arabic text transmitted via internet. *Comput. Mater. Contin.* **2021**, *66*, 195–211. [CrossRef]
7. Liu, J.; Li, Y.; Lin, M. Review of intent detection methods in the human-machine dialogue system. *J. Phys. Conf. Ser.* **2019**, *1267*, 012059. [CrossRef]
8. Al-Wesabi, F.N. Entropy-based watermarking approach for sensitive tamper detection of arabic text. *Comput. Mater. Contin.* **2021**, *67*, 3635–3648. [CrossRef]
9. Hou, Y.; Lai, Y.; Wu, Y.; Che, W.; Liu, T. Few-shot learning for multi-label intent detection. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 13036–13044.
10. Liu, H.; Zhao, S.; Zhang, X.; Zhang, F.; Sun, J.; Yu, H.; Zhang, X. A Simple Meta-learning Paradigm for Zero-shot Intent Classification with Mixture Attention Mechanism. *arXiv* **2022**, arXiv:2206.02179.
11. Siddique, A.B.; Jamour, F.; Xu, L.; Hristidis, V. Generalized zero-shot intent detection via commonsense knowledge. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual, 11–15 July 2021; pp. 1925–1929.
12. Zhang, J.G.; Hashimoto, K.; Wan, Y.; Liu, Y.; Xiong, C.; Yu, P.S. Are pretrained transformers robust in intent classification? a missing ingredient in evaluation of out-of-scope intent detection. *arXiv* **2021**, arXiv:2106.04564.
13. Zhang, H.; Xu, H.; Lin, T.E. Deep Open Intent Classification with Adaptive Decision Boundary. In Proceedings of the AAAI, Virtual, 2–9 February 2021; pp. 14374–14382.
14. Yan, G.; Fan, L.; Li, Q.; Liu, H.; Zhang, X.; Wu, X.M.; Lam, A.Y. Unknown intent detection using Gaussian mixture model with an application to zero-shot intent classification. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, online, 5–10 July 2020; pp. 1050–1060.
15. Gangadharaiah, R.; Narayanaswamy, B. Joint multiple intent detection and slot labeling for goal-oriented dialog. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019; pp. 564–569.
16. Dopierre, T.; Gravier, C.; Logerais, W. ProtAugment: Intent detection meta-learning through unsupervised diverse paraphrasing. 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), online, 1–6 August 2021; pp. 2454–2466.
17. Nigam, A.; Sahare, P.; Pandya, K. Intent detection and slots prompt in a closed-domain chatbot. In Proceedings of the 2019 IEEE 13th International Conference on Semantic Computing (ICSC), Newport Beach, CA, USA, 30 January–1 February 2019; pp. 340–343.
18. Zhou, Y.; Liu, P.; Qiu, X. Knn-contrastive learning for out-of-domain intent classification. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, 22–27 May 2022; pp. 5129–5141.
19. Büyük, O.; Erden, M.; Arslan, L.M. Leveraging the information in in-domain datasets for transformer-based intent detection. In Proceedings of the 2021 Innovations in Intelligent Systems and Applications Conference (ASYU), Elazig, Turkey, 6–8 October 2021; pp. 1–4.
20. Shen, Y.; Hsu, Y.C.; Ray, A.; Jin, H. Enhancing the generalization for Intent Classification and Out-of-Domain Detection in SLU. *arXiv* **2021**, arXiv:2106.14464.
21. Qin, L.; Xu, X.; Che, W.; Liu, T. AGIF: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. *arXiv* **2020**, arXiv:2004.10087.
22. Sreelakshmi, K.; Rafeeque, P.C.; Sreetha, S.; Gayathri, E.S. Deep bi-directional lstm network for query intent detection. *Procedia Comput. Sci.* **2018**, *143*, 939–946. [CrossRef]
23. Muthu, B.; Cb, S.; Kumar, P.M.; Kadry, S.N.; Hsu, C.H.; Sanjuan, O.; Crespo, R.G. A framework for extractive text summarization based on deep learning modified neural network classifier. *Trans. Asian Low-Resour. Lang. Inf. Process.* **2021**, *20*, 1–20. [CrossRef]
24. Mo, S.; Ye, Q.; Jiang, K.; Mo, X.; Shen, G. An improved MPPT method for photovoltaic systems based on mayfly optimization algorithm. *Energy Rep.* **2022**, *8*, 141–150. [CrossRef]

25. Coucke, A.; Saade, A.; Ball, A.; Bluche, T.; Caulier, A.; Leroy, D.; Doumouro, C.; Gisselbrecht, T.; Caltagirone, F.; Lavril, T.; et al. Snips voice platform: An embedded spoken language understanding system for private-by-design voice interfaces. *arXiv* **2018**, arXiv:1805.10190.
26. Agarwal, V.; Shivnikar, S.D.; Ghosh, S.; Arora, H.; Saini, Y. Lidsnet: A lightweight on-device intent detection model using deep siamese network. In Proceedings of the 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), Pasadena, CA, USA, 13–16 December 2022; pp. 1112–1117.