Novel Slot Detection Trong Hệ Thống Đối Thoại Hướng Nhiệm Vụ

Trần Triệu Vũ
Khoa Học Và Kĩ Thuật Thông Tin
Đại Học Công Nghệ Thông Tin
TPHCM, Việt Nam
19522539@gm.uit.edu.vn

Võ Minh Trí
Khoa Học Và Kĩ Thuật Thông Tin
Đại Học Công Nghệ Thông Tin
TPHCM, Việt Nam
19522396@gm.uit.edu.vn

Phạm Đức Thể
Khoa Học Và Kĩ Thuật Thông Tin
Đại Học Công Nghệ Thông Tin
TPHCM, Việt Nam
19522253@gm.uit.edu.vn

Nguyễn Văn Kiệt
Khoa Học Và Kĩ Thuật Thông Tin
Đại Học Công Nghệ Thông Tin
TPHCM, Việt Nam
kietnv@uit.edu.vn

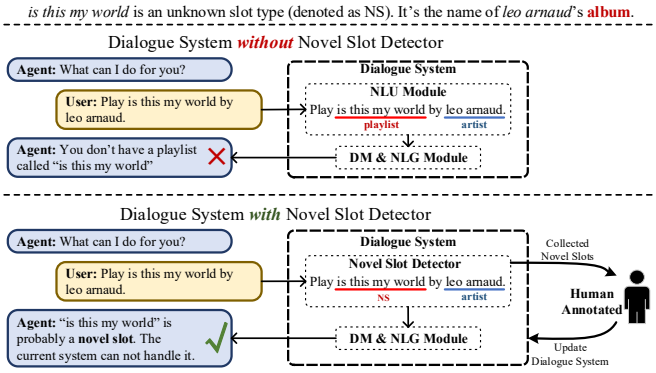
***Tóm tắt nội dung*—Các mô hình slot filling hiện tại chỉ có thể nhận diện các loại in-domain slot được định nghĩa trước dựa trên một tập slot giới hạn. Trong ứng dụng thực tế, một hệ thống đối thoại đáng tin cậy nên biết những gì nó không biết. Trong báo cáo này, chúng tôi tìm hiểu và trình bày một tác vụ mới, Novel Slot Detection (NSD), trong hệ thống đối thoại hướng nhiệm vụ. NSD nhằm mục đích khám phá các loại unknown slot hoặc out-of-domain slot để tăng cường khả năng của hệ thống đối thoại dựa trên dữ liệu huấn luyện in-domain. Bên cạnh đó, chúng tôi trình bày lại hai bộ dữ liệu và một baseline mạnh mẽ cho bài toán NSD. Cuối cùng, chúng tôi tiến hành thực nghiệm và phân tích định tính để hiểu rõ những thách thức chính và hướng phát triển trong tương lai [1].**

***Index Terms*—Novel Slot Detection, Slot Filling, Snips Dataset, ATIS Dataset, Task-Oriented Dialogue System, BiLSTM-CRF, Maximum Softmax Probability, Gaussian Discriminant Analysis, BIO tags, Token F1, Span F1, Restriction-Oriented Span Evaluation.**

I. Giới Thiệu

Slot filling đóng vai trò quan trọng để hiểu các truy vấn của người dùng trong hệ thống của các trợ lý ảo như Amazon Alexa, Apple Siri, Google Assistant, v.v. Nó nhằm mục đích xác định một chuỗi các token và trích xuất các thành phần ngữ nghĩa từ truy vấn của người dùng. Với dữ liệu huấn luyện được thu thập từ trước trên quy mô lớn, các mô hình neural-based hiện có [1]–[10] đã được áp dụng vào tác vụ slot filling và đạt được những kết quả đầy hứa hẹn.

Các mô hình slot filling hiện tại chỉ có thể nhận ra các loại thực thể được định nghĩa trước từ một tập hợp slot giới hạn, điều này là không đủ trong tình huống ứng dụng thực tế. Một mô hình slot filling đáng tin cậy không nên chỉ tập trung dự đoán được các slot được định nghĩa trước mà còn phải nhận diện được các loại *slot không xác định (unknown slot)* có tiềm năng để biết những slot chưa được định nghĩa, mà chúng tôi gọi là Novel Slot Detection (NSD). NSD đặc biệt quan trọng trong các hệ thống đã triển khai — vừa để tránh thực hiện sai hành động vừa để khám phá các loại thực thể mới tiềm năng nhằm phát triển và cải tiến trong tương lai. Ví dụ NSD được thể hiện như trong Hình 1.



Hình 1: Một ví dụ về Novel Slot Detection trong hệ thống đối thoại hướng nhiệm vụ. Nếu không có NSD, hệ thống đối thoại sẽ đưa ra phản hồi sai vì nó hiểu nhầm unknown slot "is this my world" là loại *playlist* in-domain. Ngược lại, NSD nhận ra "is this my world" là *NS* và hệ thống đưa ra phản hồi dự phòng. Trong khi đó, với đánh giá của con người trong vòng lặp, hệ thống có thể cải thiện các chức năng hoặc kỹ năng của nó.

Novel Slot (NS) được định nghĩa là loại slot mới chưa có trong tập các slot được xác định trước. NSD nhằm mục đích khám phá các loại thực thể mới hoặc *ngoài miền (out-of-domain)* tiềm năng để tăng cường khả năng của hệ thống đối thoại dựa trên dữ liệu huấn luyện *trong miền (in-domain)* đã được thu thập trước. Có hai khía cạnh trong các công trình trước đây liên quan đến NSD, nhận dạng *ngoài từ vựng (out-of-vocabulary – OOV)* [7], [11]–[16] và phát hiện ý định *ngoài miền (out-of-domain – OOD)* [17]–[20]. OOV có nghĩa là những loại slot có thể chứa một lượng lớn các giá trị slot mới

[1]https://github.com/ChestnutWYN/ACL2021-Novel-Slot-Detection

Utterance	Play	is	this	my	world	by	leo	arnaud
Slot Filling Labels	O	B-album	I-album	I-album	I-album	O	B-artist	I-artist
Novel Slot Detection Labels	O	NS	NS	NS	NS	O	B-artist	I-artist

Bảng I: So sánh giữa slot filling và NSD. Trong các nhãn NSD, chúng tôi coi "album" là một loại unknown slot nằm ngoài phạm vi của tập slot được định nghĩa trước. Trong khi đó, "artist" thuộc các loại in-domain slot vẫn được xem là tác vụ slot filling được định nghĩa ban đầu.

trong khi tập huấn luyện chỉ có được một phần nhỏ. OOV nhằm mục đích nhận diện được các giá trị slot không nhìn thấy trong tập huấn luyện, sử dụng character embedding [11], copy mechanism [21], few/zero-shot learning [13], [16], [22], transfer learning [6], [14], [23] và background knowledge [7], [24], v.v. So với nhận diện OOV, tác vụ NSD được đề xuất của tác giả tập trung vào việc phát hiện các loại *unknown slot*, không chỉ các giá trị không nhìn thấy. NSD phải đối mặt với những thách thức của cả OOV và không có ngữ nghĩa ngữ cảnh đầy đủ (xem phân tích trong Phần 6.2), làm tăng đáng kể mức độ phức tạp của tác vụ. Một công trình liên quan khác là phát hiện ý định OOD [17], [19], [25]–[28] nhằm mục đích biết khi nào một truy vấn nằm ngoài phạm vi các đề xuất có trước của hệ thống. Sự khác biệt chính là NSD phát hiện các loại unknown slot trong token-level trong khi OOD chỉ xác định các truy vấn có mục đích out-of-domain. NSD đòi hỏi sự hiểu biết sâu sắc về ngữ cảnh truy vấn và dễ bị sai lệch nhãn O (xem phân tích trong Phần 5.3.1), khiến việc xác định các loại unknown slot trong hệ thống đối thoại hướng nhiệm vụ trở nên khó khăn.

Trong báo cáo này, trước tiên chúng tôi trình bày một tác vụ mới và quan trọng là Novel Slot Detection trong hệ thống đối thoại hướng nhiệm vụ (Phần 2.2). NSD đóng một vai trò quan trọng trong việc tránh đưa ra các quyết định sai lầm và phát hiện ra các loại thực thể mới tiềm năng cho sự phát triển trong tương lai của hệ thống đối thoại hướng nhiệm vụ. Tác giả đã xây dựng hai bộ dữ liệu NSD là Snips-NSD và ATIS-NSD, dựa trên bộ dữ liệu slot filling gốc là Snips [29] và ATIS [30] (Phần 3.2). Từ góc độ ứng dụng thực tế, tác giả xem xét ba loại chiến lược xây dựng bộ dữ liệu là Replace, Mask và Remove. Replace – gán nhãn các giá trị NS với tất cả *O* trong tập training. Mask – gán nhãn là *O* và che đi các giá trị NS. Remove – là chiến lược nghiêm ngặt nhất trong đó tất cả các truy vấn chứa NS đều bị xóa. Chúng tôi đi sâu vào chi tiết của ba chiến lược xây dựng khác nhau trong Phần 3.2 và thực hiện phân tích định tính trong Phần 5.3.1. Bên cạnh đó, tác giả đề xuất hai loại chỉ số đánh giá, span-level F1 và token-level F1 trong Phần 3.4. Span F1 xem xét sự *so khớp chính xác (exact matching)* của NS span trong khi Token F1 tập trung vào *độ chính xác (accuracy)* của dự đoán trên từng từ của NS span. So sánh hiệu suất giữa hai độ đo và đề xuất một độ đo mới, *restriction-oriented span evaluation (ROSE)*, để kết hợp những ưu điểm của cả hai trong Phần 5.3.3. Sau đó, mô tả baseline cho NSD trong Phần 4. Cuối cùng, chúng tôi thực hiện các thí nghiệm toàn diện và phân tích định tính để hiểu rõ những thách thức mà các phương pháp tiếp cận hiện tại đối với NSD trong Phần 5.3 và 6.

II. Bài Toán

A. Slot Filling

Cho một câu $X = \{x_1, \ldots, x_n\}$ với n tokens, tác vụ slot filling là dự đoán tag của chuỗi tương ứng $Y = \{y_1, \ldots, y_n\}$ ở định dạng BIO, trong đó mỗi y_i có thể nhận ba loại giá trị: B-slot_type, I-slot_type và *O*, trong đó "B" và "I" là từ *đầu tiên (beginning)* và từ *trung gian (intermediate)* của một slot và "*O*" có nghĩa là từ không thuộc về bất kỳ slot nào. Ở đây, việc slot filling giả định $y_i \in y$, trong đó y biểu thị một tập hợp slot được định nghĩa trước có kích thước M. Các phương pháp hiện tại thường mô hình hóa việc slot filling như một bài toán *gán nhãn chuỗi (equence labeling)* bằng cách sử dụng RNN [2], [3], [31] hoặc pre-trained language models [5].

B. Novel Slot Detection

Chúng tôi coi dữ liệu huấn luyện là dữ liệu in-domain (IND). NSD nhằm mục đích xác định các loại unknown slot hoặc OOD thông qua dữ liệu IND trong khi gán nhãn chính xác dữ liệu in-domain. Chúng tôi biểu thị loại unknown slot là *NS* và các loại in-domain slot là IND trong các phần sau. Lưu ý rằng chúng ta không phân biệt giữa B-NS và I-NS và thống nhất chúng là *NS* vì theo kinh nghiệm, tác giả nhận thấy các mô hình hiện có hầu như không phân biệt B và I cho một loại unknown slot. Chúng tôi cung cấp một phân tích chi tiết trong Phần 5.3.3 và đưa ra một ví dụ về NSD trong Bảng I. Những thách thức trong việc nhận biết NSD đến từ hai khía cạnh, các tag *O* và các in-domain slot. Một mặt, các mô hình cần tìm hiểu thông tin thực thể để phân biệt *NS* với các tag *O*. Mặt khác, chúng yêu cầu *NS* phân biệt với các loại slot khác trong tập slot được định nghĩa trước. Chúng tôi cung cấp phân tích lỗi chi tiết trong Phần 6.1.

III. Bộ Dữ Liệu

Vì không có bộ dữ liệu NSD phù hợp, tác giả đã xây dựng hai bộ dữ liệu NSD mới [32] dựa trên hai bộ dữ liệu slot filling được sử dụng rộng rãi là Snips [29] và ATIS [30]. Đầu tiên, chúng tôi giới thiệu ngắn gọn về Snips và ATIS, sau đó sẽ trình bày cách thức xây dựng và xử lý dữ liệu một cách chi tiết của tác giả, đồng thời hiển thị thống kê về hai bộ dữ liệu Snips-NSD và ATIS-NSD. Cuối cùng, chúng tôi xác định hai độ đo đánh giá cho tác vụ NSD là Span F1 và Token F1.

A. Các Bộ Dữ Liệu Slot Filling Gốc

Snips[2] là bộ dữ liệu về đối thoại có thể tinh chỉnh để phục vụ cho các tác vụ NLU (Natural Language Understanding) khác nhau. Ban đầu nó có 13,084 câu dùng để huấn luyện

[2]https://github.com/sonos/nlu-benchmark/tree/master/2017-06-custom-intent-engines

Original Utterance		play	**is**	**this**	**my**	**world**	by	leo	arnaud
Original Slot Filling Labels		O	**B-album**	**I-album**	**I-album**	**I-album**	O	B-artist	I-artist
Strategy	**Replace**	play O	**is** **O**	**this** **O**	**my** **O**	**world** **O**	by O	leo B-artist	arnaud I-artist
	Mask	play O	**MASK** **O**	**MASK** **O**	**MASK** **O**	**MASK** **O**	by O	leo B-artist	arnaud I-artist
	Remove	- -	- -	- -	- -	- -	- -	- -	- -

Bảng II: So sánh giữa ba chiến lược xử lý trong tập training. Chúng tôi coi "album" là một loại unknown slot và "-" biểu thị câu bị xóa khỏi dữ liệu training.

	Snips	ATIS
Vocabulary Size	11,241	722
Percentage of OOV words	5.59%	0.77%
Number of Slots	39	79
Training Set Size	13,084	4,478
Development Set Size	700	500
Testing Set Size	700	893

Bảng III: Thống kê bộ dữ liệu ATIS và Snips.

(train utterances), 700 câu dùng để phát triển (dev utterances) và 700 câu dùng để kiểm thử (test utterance). ATIS[3] chứa các bản ghi âm của những người đặt chỗ chuyến bay. Ban đầu, nó có 4,478 câu huấn luyện, 500 câu phát triển và 893 câu kiểm thử. Thống kê đầy đủ được thể hiện trong Bảng II. Lưu ý rằng tập từ vựng chỉ bao gồm các từ trong tập train, các từ thuộc tập test không tồn tại trong tập từ vựng được gọi là các từ OOV. *Percentage of OOV words* thể hiện phần trăm các từ OOV trong tập test.

B. Xây Dựng và Xử Lý Dữ Liệu

Đối với tập dữ liệu Snips và ATIS, tác giả giữ lại một số loại slot trong training là unknown và tích hợp chúng trở lại trong quá trình testing (theo [17], [33], [34]). Tác giả chọn ngẫu nhiên một phần các loại slot trong Snips và ATIS trở thành các unknown slot (5%, 15% và 30%). Lưu ý rằng chia train/val/test ban đầu đã được cố định. Xem xét sự mất cân bằng dữ liệu, tác giả thực hiện lấy mẫu có trọng số trong đó xác suất được chọn có liên quan đến số lượng các ví dụ về slot tương tự như [17]. Để tránh tính ngẫu nhiên của kết quả thử nghiệm, tác giả báo cáo kết quả trung bình trên 10 lần chạy.

Sau khi tác giả chọn các loại unknown slot, một vấn đề quan trọng là làm thế nào để xử lý các câu bao gồm các loại unknown slot trong tập training. Để phát hiện OOD, tác giả chỉ cần xóa những câu này trong tập training và tập validation. Tuy nhiên, đối với NSD, một câu có thể chứa cả các in-domain slot và các unknown slot, điều này không đơn giản để giải quyết các unknown slot ở token level. Chúng ta cần cân bằng hiệu suất nhận dạng các unknown slot và các in-domain slot. Do đó, tác giả đã đề xuất ba chiến lược xử lý khác nhau như sau: (1) **Replace**: gán nhãn *O* cho tất cả các giá trị unknown slot trong tập training và các giá trị ban đầu không thay đổi. (2) **Mask**: gán nhãn *O* cho tất cả các giá trị unknown slot và che các giá trị slot này bằng một token đặc biệt – *MASK*. (3) **Remove**: Tất cả các câu chứa các unknown slot được loại bỏ trực tiếp.

Các ví dụ về ba chiến lược trên được trình bày trong Bảng II. Đối với tập val và tập test, tác giả chỉ gán nhãn *NS* với tất cả các giá trị unknown slot và giữ nguyên nhãn in-domain. Lưu ý rằng các tag *NS* chỉ tồn tại trong tập val và tập test, không tồn tại trong tập training. Bên cạnh đó, tác giả giữ cố định các in-domain slot ban đầu để đánh giá hiệu suất của các *NS* và các in-domain slot. Mục đích là mô phỏng tình huống thực tế mà chúng ta khó có thể biết được đâu là những unknown slot. Ba chiến lược này đều có ý nghĩa thiết thực của nó. So với các chiến lược khác, Remove là chiến lược phù hợp nhất cho các tình huống trong thực tế. Trong tình huống thực tế, hệ thống đối thoại trước tiên huấn luyện trong tập dữ liệu được gán nhãn bởi các *người gán nhãn (annotators)* và sau đó áp dụng cho ứng dụng thực tế. Trong quá trình tương tác với người dùng thực, các loại *NS* dần xuất hiện. Do đó, tác giả cho rằng tập training không chứa các câu *NS* tiềm năng. Nói cách khác, Remove là chiến lược phù hợp nhất cho NSD trong các ứng dụng thực tế. Hơn nữa, Phần 5.3.1 cho thấy Remove hoạt động tốt nhất trong khi các phần khác mô hình bị *sai lệch (bias)* nghiêm trọng bởi các tag *O*. Do đó, chúng tôi áp dụng Remove làm chiến lược chính.

C. Thống Kê Về Tập Dữ Liệu NSD Mới

Bảng IV cho thấy số liệu thống kê chi tiết của Snips-NSD-15% được xây dựng bằng chiến lược Remove, trong đó tác giả chọn 15% các slot trong dữ liệu training làm unknown slot.[4] Kết hợp Bảng III và Bảng IV, chúng ta có thể thấy chiến lược Remove loại bỏ 28.70% các truy vấn trong tập Snips training gốc, do đó tăng *percentage of OOV word* từ 5.95% lên 8.51%. Và các giá trị unknown slot chiếm 12.29% tổng giá trị slot trong tập test.

D. Độ Đo

Tác vụ slot filling truyền thống sử dụng Span F1[5] để đánh giá. Span F1 xem xét *so khớp span chính xác (exact span matching)* của một unknown slot. Tuy nhiên, tác giả nhận thấy rằng độ đo này quá nghiêm ngặt đối với các mô hình NSD. Trong ứng dụng thực tế, chúng ta chỉ cần khai thác thô các phần của các từ có unknown slot, sau đó gửi các truy vấn có các unknown slot token tiềm năng đến các annotator, điều này

[3]https://github.com/yvchen/JointSLU/tree/master/data

[4]Vì các tỷ lệ khác nhau của các unknown slot có số liệu thống kê khác nhau, ở đây chúng tôi chỉ hiển thị kết quả của Snips-NSD-15% cho ngắn gọn.

[5]https://www.clips.uantwerpen.be/conll2000/chunking/conlleval.txt

Snips-NSD-15%	Train	Val	Test
number of in-domain slots	33	33	33
number of unknown slots	6	6	6
percentage of OOV words	-	-	8.51%
number of queries	9,329	700	700
number of queries including unknown slots	0	192	192
number of slot values	23,176	1,794	1,790
number of unknown slot values	0	210	220

Bảng IV: Thống kê chi tiết của bộ dữ liệu Snips-NSD-15%.

đã giảm bớt lao động và nâng cao hiệu quả. Do đó, tác giả xác định một độ đo hợp lý hơn là Token F1, tập trung vào *đối sánh cấp độ từ (word-level matching)* của một *NS* span, tác giả cũng đề xuất một độ đo mới, *Restriction-Oriented Span Evaluation (ROSE)*, để so sánh công bằng trong Phần 5.3.3.

IV. Cấu Trúc Mô Hình

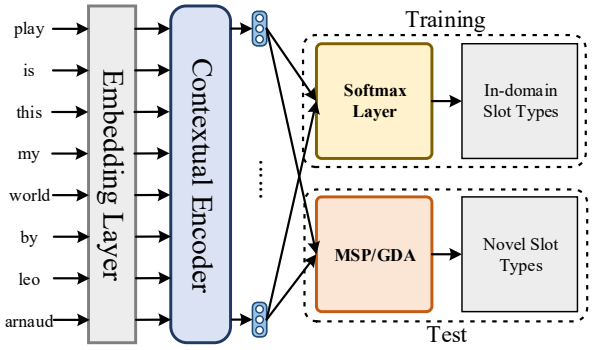
Trong phần này, chúng tôi giới thiệu các mô hình NSD được đề xuất trong bài báo của tác giả [32] và minh họa sự khác biệt giữa các phương pháp tiếp cận song song khác nhau trong giai đoạn training và test.

A. Tổng quan Framework

Cấu trúc tổng thể của mô hình được thể hiện trong Hình 2. Trong giai đoạn training, thực hiện huấn luyện một *multiple-class classifier* hoặc *binary classifier* sử dụng các mục tiêu huấn luyện khác nhau. Sử dụng lớp embedding BERT-Large và mô hình BiLSTM-CRF để trích xuất đặc trưng token level. Sau đó trong giai đoạn test, sử dụng *neural multiple classifier* điển hình để dự đoán các nhãn in-domain. Trong khi đó, sử dụng các detection algorithm là MSP hoặc GDA để tìm ra các NS token. Cuối cùng, ghi đè các nhãn slot token vừa tìm được là *NS*. Về *training objectives*, *detection algorithms* và *distance strategies*, chúng tôi so sánh các biến thể khác nhau như sau.

Training objective: Đối với các in-domain slot, tác giả đề xuất hai training objective. Multiple classifier đề cập đến mục tiêu cài đặt slot filling truyền thống, thực hiện token-level multiple classifications trên các BIO tag [35] kết hợp với các slot khác nhau. Binary classifier hợp nhất tất cả các tag non-O thành một lớp và mô hình thực hiện token-level binary classification để phân loại O hoặc non-O trên chuỗi. Lưu ý rằng trong giai đoạn test, đối với dự đoán in-domain, tác giả đều sử dụng multiple classifier. Trong khi đối với NSD, tác giả sử dụng multiple classifier hoặc binary classifier, hoặc cả hai. Trong Bảng V, chúng tôi thực hiện cài đặt lại mô hình với training objective là multiple.

Detection algorithm: MSP và GDA là các thuật toán detection trong giai đoạn test để xác định các out-of-domain (novel slot). MSP (Maximum Softmax Probability) [25] áp dụng một *threshold (ngưỡng)* cho MSP, nếu mức tối đa giảm xuống dưới threshold, token sẽ được dự đoán là *NS*. GDA (Gaussian Discriminant Analysis)) [19] là sự lựa chọn tốt cho trường hợp tác vụ phân loại có biến đầu vào liên tục và rơi vào phân phối Gaussian, chúng ta có thể hiểu GDA là một



Hình 2: Kiến trúc tổng quát của bài toán

bộ phân loại dựa trên khoảng cách chung để phát hiện out-of-domain trong không gian Euclide. Đây là phương pháp được chúng tôi sử dụng trong việc cài đặt lại mô hình.

Distance strategy: Phương thức GDA xây dựng dựa trên khoảng cách giữa mục tiêu và mỗi cụm biểu diễn slot. Trong GDA gốc, khi khoảng cách tối thiểu lớn hơn một threshold nhất định nó được dự đoán là các NS. Tác giả đề xuất một cách mới có tên là *Difference*, cách này được hiểu là sử dụng khoảng cách tối đa trừ đi khoảng cách tối thiểu, khi giá trị chênh lệch vừa tính ra nhỏ hơn threshold, nó được dự đoán là NS. Cả hai cách trên threshold của chúng đều có được bằng cách tối ưu hóa các chỉ số NSD trên tập validation. Ở Bảng V chúng tôi thực hiện cài đặt lại mô hình với distance stategy là minimum tức theo GDA mặc định.

V. Thực Nghiệm và Phân Tích Kết Quả

A. Chi Tiết Triển Khai

Chúng tôi dựa trên mã nguồn do tác giả cung cấp để cài đặt lại mô hình trên bộ dữ liệu Snips, còn bộ dữ liệu ATIS (đã được gán nhãn) không được tác giả public. Thực hiện xây dựng lại mô hình với các thông số sau: sử dụng lại các cài đặt cho mô hình pre-trained Bert-large-uncased để embed tokens có 24 layers, 1024 hidden states, 16 heads và 336M parameters. Hidden size cho BiLSTM layers được đặt thành 128. Adam được sử dụng để tối ưu hóa với learning rate ban đầu là 2e-5. Giá trị dropout được cố định là 0.5 và batch size là 64. Huấn luyện mô hình trên dữ liệu có nhãn in-domain. Giai đoạn training có cài đặt early stopping với patience bằng 10. Chúng tôi sử dụng F1-score tốt nhất trên tập validation để tính toán theshold GDA một cách phù hợp. Giai đoạn training mô hình của kéo dài khoảng 30 phút trên GPU Tesla T4 duy nhất (bộ nhớ 16 GB) trên môi trường Google Colab Pro.

B. Kết Quả Chính

Bảng V là kết quả thực nghiệm với mô hình tốt nhất theo phân tích của tác giả: sử dụng detection method là GDA, training objective là multiple và distance stragy là minimum trên bộ dữ liệu Snips-NSD được xây dựng bằng chiến lược Remove. Kết quả được trình bày bao gồm Span F1 cho IND và Span F1, Token F1 cho NSD tương ứng với bộ dữ liệu có tỷ lệ 5%, 15%, 30% là unknown slot. Ảnh hưởng của tỷ lệ unknown slot này được mô tả trong 5.3.2.

Model			5%			15%			30%		
			IND	NSD		IND	NSD		IND	NSD	
detection method	objective	distance stragy	Span F1	Span F1	Token F1	Span F1	Span F1	Token F1	Span F1	Span F1	Token F1
GDA	multiple	minimum	92.33	28.27	58.09	85.21	18.46	42.11	87.32	17.21	42.14

Bảng V: Kết quả thực nghiệm trên Snips-NSD 5%, 15%, 30%.

Strategy	5%			15%			30%		
	IND	NSD		IND	NSD		IND	NSD	
	Span F1	Span F1	Token F1	Span F1	Span F1	Token F1	Span F1	Span F1	Token F1
Replace	94.52	1.93	5.27	94.33	0.66	2.29	94.02	0.27	0.82
Mask	90.08	23.10	37.91	86.52	25.07	45.92	83.37	32.14	50.68
Remove	93.10	**31.67**	**46.97**	90.18	**32.19**	**53.75**	86.26	**38.64**	**55.24**

Bảng VI: So sánh sự khác nhau giữa các chiến lược xử lý dữ liệu trên Snips-NSD sử dụng mô hình GDA+Multiple+Minimum.

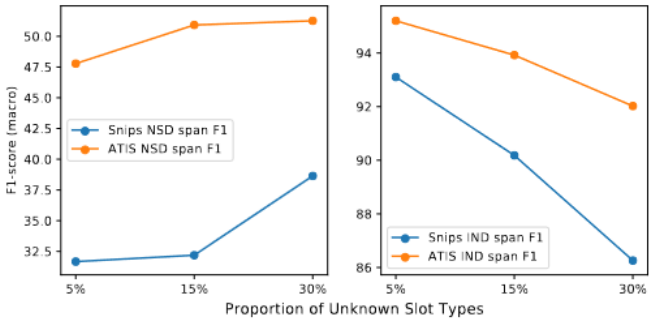
C. Phân Tích Định Tính

Phần này được trình bày dựa trên kết quả của tác giả.

1) Sự ảnh hưởng của các chiến lược sử lý dữ liệu: Bảng VI trình bày các chỉ số IND và NSD của ba chiến lược xử lý bộ dữ liệu khác nhau trên Snips-NSD sử dụng cùng một mô hình GDA+Multiple+Minimum. Trong phần này, tác giả đi sâu vào phân tích ảnh hưởng của các chiến lược xử lý dữ liệu khác nhau. Kết quả cho thấy chiến lược Replace có hiệu suất kém trong NSD, điều này chứng tỏ việc gán nhãn các unknown slot bởi các tag *O* sẽ gây hiểu lầm nghiêm trọng cho mô hình. Chiến lược Mask và Remove hợp lý hơn vì chúng loại bỏ các unknown slot khỏi dữ liệu training. Sự khác biệt chính của chúng là Mask chỉ xóa thông tin token-level, trong khi Remove thậm chí loại bỏ thông tin theo ngữ cảnh. Đối với NSD trong tất cả các bộ dữ liệu, Remove đạt được hiệu suất tốt hơn đáng kể trên cả Token F1 và Span F1 so với Mask 9.06%(5%), 7.83%(15%) và 4.56%(30%) trên Token F1 và 8.57%(5%), 7.12%(15%) và 6.5%(30%) trên Span F1. Tác giả cho rằng ngữ cảnh còn lại vẫn gây hiểu nhầm ngay cả khi các NS token không được huấn luyện trực tiếp trong chiến lược Mask. Bên cạnh đó, Mask không phù hợp với dữ liệu NSD thực tế. Nhìn chung, Remove vẫn là chiến lược phù hợp nhất cho bài toán NSD trong các ứng dụng thực tế.

2) Sự ảnh hưởng của tỷ lệ các loại unknown slot trong bộ dữ liệu: Hình 3 trình bày ảnh hưởng của tỷ lệ các loại unknown slot bằng cách sử dụng chiến lược Remove trong GDA+Multiple+Minimum. Kết quả cho thấy rằng với sự gia tăng của tỷ lệ các loại unknown slot, NSD F1 score được cải thiện trong khi IND F1 score giảm. Tác giả cho rằng ít loại in-domain slot hơn giúp mô hình phân biệt các unknown slot với các IND slot tốt hơn nên NSD F1 score sẽ được cải thiện. Tuy nhiên, đối với tác vụ phát hiện in-domain slot, vì chiến lược Remove xóa tất cả các câu chứa unknown slot trong dữ liệu training, các mô hình nhận về thiếu hụt ngữ cảnh so với trước đó nên hiệu suất nhận ra IND slot giảm đi và IND F1 score giảm.

3) Độ đo đánh giá mới: ROSE: Các kết quả trước đó đã cho thấy Span F1 thấp hơn nhiều so với Token F1. Nguyên nhân là do bản thân Span F1 là một độ đo nghiêm ngặt, nó yêu cầu mô hình phải dự đoán chính xác tất cả các *NS* token và ranh giới tương ứng. Đây là một khó khăn khi các mô hình



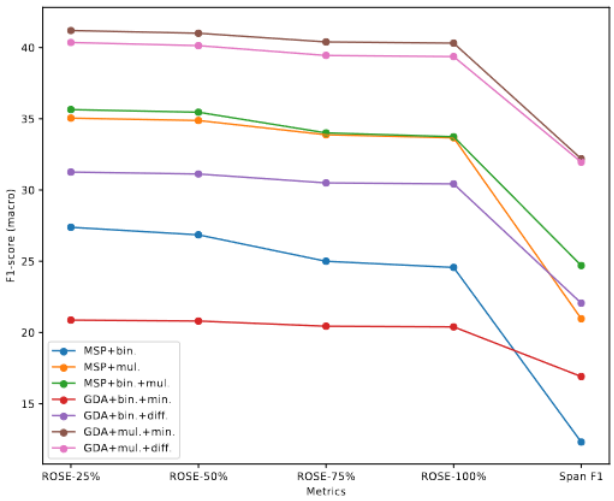
Hình 3: Ảnh hưởng của tỷ lệ các loại unknown slot trong bộ dữ liệu.

	GDA+mul.+min.	MSP+bin.+mul.
ROSE-mean	40.73	34.71
ROSE-100%	40.39	33.74
ROSE-50%	41.00	35.46

Bảng VII: Độ đo ROSE trên Snips-NSD 15% sử dụng mô hình GDA+Multiple+Minimum và MSP+Binary+Multiple.

NSD vốn hạn chế về thông tin được giám sát. Trong khi đó, trên thực tế, các mô hình NSD chỉ cần đánh dấu đúng một số token trong span của các NS và gửi tổng chuỗi chứa các *NS* token đó trở lại con người, hoàn toàn có thể chấp nhận được một số lượng nhỏ thiếu sót hoặc đánh giá sai token. Do đó, để đáp ứng cho một tình huống NSD hợp lý, tác giả đề xuất một độ đo mới, *restriction-oriented span evaluation (ROSE)*, để đánh giá hiệu suất dự đoán span dưới các hạn chế khác nhau. Đầu tiên, nó không tính lỗi các trường hợp dự đoán các token vượt quá span. Sau đó, nó coi một span là đúng khi số lượng token được dự đoán chính xác lớn hơn một tỷ lệ p có thể tinh chỉnh của độ dài span. Chúng tôi lấy trung bình của ROSE score và Span F1 gốc để tránh mô hình thu được kết quả vượt trội thông qua dự đoán quá lâu. Kết quả sử dụng Snips với 15% NS được thể hiện trong Hình 4. Khi mức p tăng lên, các độ đo có xu hướng giảm. Điều đó chỉ ra rằng mô hình chủ yếu chỉ có thể xác định một nửa số token trong các span. Tuy nhiên, để đánh giá một cách toàn diện, tác giả đã xác định ROSE-mean, cụ thể là giá trị trung bình của ROSE-25%, ROSE-50%, ROSE-75% và ROSE-100%. Kết quả thực nghiệm trên hai mô hình mà tác giả đề xuất được trình bày trong Bảng VII.

4) Phân tích theo từng loại unknown slot: Để phân tích mối quan hệ giữa hiệu suất NSD và một slot cụ thể, tác giả tính toán các độ đo token và span coi mỗi loại slot duy nhất là một unknown slot và hiển thị kết quả của top 5 cao nhất và top 5 thấp nhất trong Bảng VIII. Tác giả nhận thấy rằng các slot có hiệu suất tốt hơn thường chiếm tỷ lệ lớn trong tập



Hình 4: Ảnh hưởng của các mức độ hạn chế khác nhau.

Type		Proportion(%)	Span Length	Token F1	Span F1
top 5	Object_name	21.42	3.71	55.64	20.82
	TimeRange	15.29	2.35	53.65	30.15
	Entity_name	23.14	3.09	48.56	22.83
	Music_item	14.86	1.05	46.23	34.59
	Artist	15.29	2.05	45.26	26.36
bottom 5	City	8.57	1.32	18.72	15.85
	Country	6.29	1.57	14.19	11.11
	State	5.54	1.10	13.55	10.83
	Best_rating	6.14	1.00	11.04	11.04
	Year	3.43	1.00	10.24	10.24

Bảng VIII: Kết quả theo từng loại unknown slot.

dữ liệu, chẳng hạn như *Object_name* hoặc *Entity_name*. Hoặc chúng có thể có xu hướng mang không gian giá trị lớn, chẳng hạn như *TimeRange*, *Music_item* hoặc *Artist*. Những đặc điểm này giúp cho việc biểu diễn ngữ nghĩa của các slot này được phân phối trên một khu vực rộng lớn hơn là phân cụm chúng lại với nhau. Tác giả cho rằng phân phối này hợp lý hơn vì trong một tình huống ứng dụng thực tế, các NS rất đa dạng và phân phối của nó có xu hướng phân tán. Hiệu suất trên các loại này cho thấy rằng các mô hình NSD mà chúng tôi đề xuất có khả năng tổng quát hóa tốt hơn một thiết lập dữ liệu hợp lý.

5) Phân tích mối quan hệ của các loại unknown slot: Để tìm hiểu ảnh hưởng của mối quan hệ giữa các inter-slot trên NSD, tác giả đã tiến hành các thí nghiệm trong đó hai loại được trộn lẫn làm NS. Một số kết quả được thể hiện trong Bảng IX. Trong năm kiểu kết hợp trong bảng, *Object_name* là một open vocabulary slot với nhiều giá trị và chứa nhiều OOV tokens, *TimeRange* và *Party_size* thường chứa số, *City* và *State* thường giống nhau về ngữ nghĩa và ngữ cảnh. Tác giả nhận thấy rằng khi các loại khác kết hợp với *Object_name*, hiệu suất NSD thường được duy trì gần với việc coi *Object_name* như một NS khác biệt. Một mặt, nguyên nhân là do tỷ lệ của các loại slot khác trong tập dữ liệu tương đối nhỏ, do đó tác động lên tổng thể nhỏ hơn. Mặt khác, do phạm vi phân bố ngữ nghĩa lớn của open vocabulary slot, tiềm ẩn mối quan hệ bao hàm các loại slot khác, do đó việc trộn lẫn các loại slot có xu hướng ảnh hưởng nhẹ đến hiệu suất NSD. Chúng

Type 1	Type 2	Token F1	Span F1
Object_name	-	55.64	20.82
TimeRange	-	53.65	30.15
Party_size_number	-	33.44	28.57
City	-	18.72	15.85
State	-	13.55	10.83
Object_name	TimeRange	53.88	23.37
Object_name	Party_size_number	52.81	22.35
Object_name	City	57.92	21.42
Object_name	State	56.32	19.27
TimeRange	Party_size_number	71.27*	51.03*
City	State	29.33*	27.14*

Bảng IX: Kết quả phân sự kết hợp các loại unknown slot. * thể hiện hiệu suất NSD của sự kết hợp của hai loại unknown slot tốt hơn đáng kể so với mỗi unknown slot đơn lẻ.

NSD error proportion(%)	O	Open vocabulary slots	Other slots	Sum
Prediction is NS	17.79	18.84	9.07	45.70
Target is NS	18.47	7.54	28.29	54.30
Sum	36.26	26.38	37.36	100.00

Bảng X: Tỷ lệ tương đối của một số loại lỗi.

Error type	NS	Example
NS to O	movie_name (m_name)	text: when will paris by night aired true: O O B-m_name **I-m_name** I-m_name O predict: O O NS **O** NS O
NS to open slot	album	text: play the insoc ep true: O **B-album I-album** I-album predict: O **B-object_name I-object_name** NS
NS to other slot	artist	text: play kurt cobain ballad tunes true: O **B-artist I-artist** B-music_item O predict: O **B-genre I-genre** B-music_item O
O to NS	artist	text: the workout playlist needs more chris cross true: O B-playlist O O **O** B-artist I-artist predict: O B-playlist O O **NS** NS NS
open slots to NS	object_type	text: tell me the actors of the saga awards true: O O O **B-object_name** O O B-object_type O predict: O O O **NS** O O NS O
orther slots to NS	city	text: what is the weather os east portal ks true: O O O O O B-city I-city **B-state** predict: O O O O O NS NS **NS**

Bảng XI: Các trường hợp lỗi trong dự đoán NSD.

tôi cũng nhận thấy rằng sự kết hợp thích hợp có thể cải thiện đáng kể hiệu quả của NSD. Chẳng hạn như *TimeRange* với *Party_size_number* hoặc *City* với *State*, điều này chỉ ra khi các NS tương tự như in-domain slot, mô hình có xu hướng dự đoán NS là một slot tương tự, điều này dẫn đến lỗi. Khi cả hai đều được coi là NS, những lỗi này có thể được giảm thiểu.

VI. Phân Tích Lỗi và Thách Thức

Theo kinh nghiệm, tác giả chia các lỗi thành ba loại. Mỗi loại bao gồm hai khía cạnh, tương ứng với precision và recall NSD. Tác giả sử dụng tập dữ liệu Snips với 5% NS trên mô hình GDA+multiple+minimum để trình bày tỷ lệ tương đối một số loại lỗi qua Bảng X. Đối với mỗi loại lỗi, tác giả trình bày một ví dụ qua Bảng XI để mô tả các đặc điểm và phân tích nguyên nhân. Cuối cùng, tác giả đi sâu vào xác định những thách thức chính sau đó đề xuất các giải pháp khả thi cho các nghiên cứu sau này.

A. Phân Tích Lỗi

Tag *O*: Chiếm số lượng lớn nhất và được phân phối rộng rãi nhất trong tập dữ liệu và nó thường đề cập đến các token có chức năng độc lập. Do đó, khi xác định, mô hình dễ bị nhầm lẫn với các loại khác và sự nhầm lẫn này càng nghiêm trọng hơn đối với các NS, thứ không được *học giám sát (supervised learning)*. Tác giả nhận thấy rằng các token có nhãn *O* được phát hiện là NS thường tồn tại gần các span và các từ chức năng trong span được gán nhãn là NS có xác suất được dự đoán là *O*. Tác giả cho rằng loại lỗi này có liên quan đến ngữ cảnh. Mặc dù chiến lược xử lý Remove có thể làm giảm hiệu quả sự sai lệch của *O* đối với các NS, tag *O* vẫn sẽ bị ảnh hưởng bởi thông tin ngữ cảnh của các in-domain slot khác.

Open Vocabulary Slots: Tác giả nhận thấy rằng một số lượng lớn các NS token bị đánh giá nhầm là open vocabulary slots, trong khi tình huống ngược lại ít có khả năng xảy ra hơn nhiều. Điều này chỉ ra rằng trong Snips, các open vocabulary slots có xu hướng chồng lên nhau hoặc chứa hầu hết các slot khác về mặt ngữ nghĩa. Ngay cả trong các tác vụ slot filling truyền thống, các open vocabulary slots thường bị nhầm lẫn với các slot khác. Chúng tôi chứng minh giả thuyết này trong phân tích. Phần 5.3.5 cho thấy NSD hoạt động tốt hơn khi các open vocabulary slots được coi như các NS và Phần 5.3.4 cho thấy rằng không có thay đổi đáng kể về hiệu suất khi các open vocabulary slots được trộn lẫn với một số slot tập trung về ngữ nghĩa. Nguyên nhân của vấn đề này là định nghĩa của bộ dữ liệu không hợp lý. Các slot có phạm vi giá trị lớn khó có thể giúp trợ lý ảo đưa ra câu trả lời thích hợp và thông tin được giám sát của các slot này thường không đầy đủ.

Similar Slots: Ngoại trừ hai trường hợp được đề cập ở trên, dự đoán các NS như các in-domain slots khác là loại lỗi phổ biến nhất, trong đó các similar slots chiếm một phần lớn trong số đó. Do sự trùng lặp giữa các từ vựng hoặc ngữ cảnh tương tự được chia sẻ, mô hình thường có xu hướng quá tự tin để dự đoán các nhãn similar slots, chúng tôi phân tích hiện tượng trong Bảng IX, khi các loại tương tự được coi như một slot mới tại cùng thời điểm, hiệu quả NSD sẽ tăng lên đáng kể. Chúng tôi sử dụng phương pháp GDA, so với phương pháp MSP truyền thống, để sử dụng đầy đủ các đặc trưng dữ liệu và giảm bớt lỗi này.

B. Thách Thức

Dựa trên phân tích trên, chúng tôi tóm tắt những thách thức hiện tại mà tác vụ NSD phải đối mặt:

Function tokens: Các mạo từ, giới từ, v.v. hoạt động như các từ liên kết trong một chuỗi. Nó thường được gán nhãn loại *O*, nhưng cũng được tìm thấy trong một số long-span slots, chẳng hạn như Movie_name. Nó có thể dẫn đến nhầm lẫn giữa *O* và NS khi loại slot này là mục tiêu của NSD.

Insufficient context: Việc phát hiện đúng slot thường phụ thuộc vào ngữ cảnh và thông tin được giám sát, trong khi đó các phần này lại bị thiếu đối với các NS. Các mô hình chỉ có thể tiến hành NSD bằng các token sử dụng các embeddings hoặc representations từ tập huấn luyện trong các ngữ cảnh khác với ngữ cảnh trong tập kiểm thử, điều này có thể dẫn đến sai lệch trong mô hình ngữ nghĩa của NS.

Dependencies between slots: Có một số sự chồng chéo về ngữ nghĩa hoặc mối quan hệ bao hàm trong định nghĩa slot của bộ dữ liệu điểm chuẩn slot filling hiện tại. Do đó, các đặc điểm ngữ nghĩa không đủ phân biệt giữa các slot và vì vậy một số token ngoại lệ trong các in-domain slot dễ bị nhầm lẫn với các NS.

Open vocabulary slots: Là một slot đặc biệt, định nghĩa của nó thường mang tính vĩ mô và có thể chia nhỏ hơn nữa, phạm vi giá trị rộng. Sự phân bố đại diện cho các Open vocabulary slots có xu hướng phân tán và không đồng đều, có thể gây hiểu lầm cho NSD.

C. Hướng phát triển

Đối với tag *O*, một giải pháp khả thi là sử dụng mô hình nhị phân để hỗ trợ xác định giữa các function token *O* và non-*O*, chúng tôi trình bày một phương pháp đơn giản trong báo cáo này và tiếp tục tối ưu hóa cho các nghiên cứu trong tương lai. Sau đó, để tách biệt sự phụ thuộc giữa các slot, điều quan trọng là phải tìm hiểu thêm các cách phân biệt khác đối với dữ liệu in-domain, sử dụng phương pháp *học đối chiếu (contrastive learning)* hoặc *mạng nguyên mẫu (prototypical network*) được kỳ vọng sẽ hữu ích. Sự kết hợp thích ứng và cải tiến các phương pháp liên quan với các tác vụ NSD cũng là một hướng nghiên cứu quan trọng trong tương lai.

VII. Công Trình Liên Quan

OOV Recognition OOV nhằm mục đích nhận ra các giá trị slot không nhìn thấy trong tập huấn luyện cho các loại slot được định nghĩa trước, sử dụng character embedding [11], copy mechanism [21], few/zero-shot learning [13], [22], transfer learning [14], [23] và background knowledge [7], [24] v.v. Tác giả đề xuất rằng tác vụ NSD tập trung vào việc phát hiện các loại unknown slot, không phải chỉ riêng các giá trị không nhìn thấy.

OOD Intent Detection [17], [19], [26] nhằm mục đích biết khi nào một truy vấn nằm ngoài phạm vi các hướng đề xuất đã có và định nghĩa trước của hệ thống. Trước tiên chúng học các biểu diễn discriminative intent thông qua dữ liệu in-domain (IND), sau đó sử dụng các thuật toán detection, chẳng hạn như Maximum Softmax Probability [25], Local Outlier Factor [17], Gaussian Discriminant Analysis [19] để tính toán sự giống nhau của các đặt trưng giữa mẫu OOD và mẫu IND. So với NSD, sự khác biệt chính là NSD phát hiện các loại unknown slot trong token level trong khi OOD intent detection chỉ xác định các truy vấn OOD trong sentence-level.

VIII. Kết Luận

Trong báo cáo này, chúng tôi đã tìm hiểu và trình bày lại một tác vụ mới là Novel Slot Detection (NSD) thông qua bộ dữ liệu Snips-NSD và các phương pháp để giải quyết nó. Chúng tôi cũng đã tìm hiểu được một số mô hình mạnh mẽ để xử lý bài toán NSD và chạy lại thành công mô hình tốt nhất mà tác giả đã đề xuất với các kết quả đạt được trên bộ Snips-NSD-5% như sau: 92.33% (IND–SpanF1), 28.27%(NSD–SpanF1), 58.09%(NSD–TokenF1) (kết quả đầy đủ được thể hiện tại Bảng V). Chúng tôi nhận thấy rằng NSD–TokenF1 (5%) và

IND–SpanF1 (30%) cho kết quả cao hơn từ 1.06% – 11.12% so với kết quả của tác giả, còn lại trên các kết quả khác đều thấp hơn từ 0.77% – 21.43% . Cuối cùng, chúng tôi biết được các thách thức chính của NSD thông qua nhiều thí nghiệm của tác giả và các giải pháp khả thi cho các nghiên cứu trong tương lai.

TÀI LIỆU

[1] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu *et al.*, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, 2014.

[2] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," *arXiv preprint arXiv:1609.01454*, 2016.

[3] C.-W. Goo, G. Gao, Y.-K. Hsu, C.-L. Huo, T.-C. Chen, K.-W. Hsu, and Y.-N. Chen, "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, pp. 753–757.

[4] E. Haihong, P. Niu, Z. Chen, and M. Song, "A novel bi-directional interrelated model for joint intent detection and slot filling," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5467–5471.

[5] Q. Chen, Z. Zhuo, and W. Wang, "Bert for joint intent classification and slot filling," *arXiv preprint arXiv:1902.10909*, 2019.

[6] K. He, W. Xu, and Y. Yan, "Multi-level cross-lingual transfer learning with language shared and specific knowledge for spoken language understanding," *IEEE Access*, vol. 8, pp. 29 407–29 416, 2020.

[7] K. He, Y. Yan, and W. Xu, "Learning to tag oov tokens by integrating contextual representation and background knowledge," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 619–624.

[8] Y. Yan, K. He, H. Xu, S. Liu, F. Meng, M. Hu, and W. Xu, "Adversarial semantic decoupling for recognizing open-vocabulary slots," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6070–6075.

[9] S. Louvan and B. Magnini, "Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey," in *28th International Conference on Computational Linguistics*, 2020, pp. 480–496.

[10] K. He, S. Lei, Y. Yang, H. Jiang, and Z. Wang, "Syntactic graph convolutional network for spoken language understanding," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 2728–2738.

[11] D. Liang, W. Xu, and Y. Zhao, "Combining word-level and character-level representations for relation classification of informal text," in *Proceedings of the 2nd Workshop on Representation Learning for NLP*. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 43–47. [Online]. Available: https://aclanthology.org/W17-2606

[12] L. Zhao and Z. Feng, "Improving slot filling in spoken language understanding with joint pointer and attention," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 426–431.

[13] Z. Hu, T. Chen, K.-W. Chang, and Y. Sun, "Few-shot representation learning for out-of-vocabulary words," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 4102–4112. [Online]. Available: https://aclanthology.org/P19-1402

[14] K. He, Y. Yan, H. Xu, S. Liu, Z. Liu, and W. Xu, "Learning label-relational output structure for adaptive sequence labeling," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[15] Y. Yan, K. He, H. Xu, S. Liu, F. Meng, M. Hu, and W. Xu, "Adversarial semantic decoupling for recognizing open-vocabulary slots," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 6070–6075. [Online]. Available: https://aclanthology.org/2020.emnlp-main.490

[16] K. He, J. Zhang, Y. Yan, W. Xu, C. Niu, and J. Zhou, "Contrastive zero-shot learning for cross-domain slot filling with adversarial attack," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 1461–1467.

[17] T.-E. Lin and H. Xu, "Deep unknown intent detection with margin loss," *arXiv preprint arXiv:1906.00434*, 2019.

[18] S. Larson, A. Mahendran, J. J. Peper, C. Clarke, A. Lee, P. Hill, J. K. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang *et al.*, "An evaluation dataset for intent classification and out-of-scope prediction," *arXiv preprint arXiv:1909.02027*, 2019.

[19] H. Xu, K. He, Y. Yan, S. Liu, Z. Liu, and W. Xu, "A deep generative distance-based classifier for out-of-domain detection with mahalanobis space," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 1452–1460.

[20] Z. Zeng, H. Xu, K. He, Y. Yan, S. Liu, Z. Liu, and W. Xu, "Adversarial generative distance-based classifier for robust out-of-domain detection," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 7658–7662.

[21] L. Zhao and Z. Feng, "Improving slot filling in spoken language understanding with joint pointer and attention," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 426–431. [Online]. Available: https://aclanthology.org/P18-2068

[22] D. J. Shah, R. Gupta, A. A. Fayazi, and D. Hakkani-Tur, "Robust zero-shot cross-domain slot filling with example values," *arXiv preprint arXiv:1906.06870*, 2019.

[23] L. Chen and A. Moschitti, "Transfer learning for sequence labeling using source model and target data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 6260–6267.

[24] B. Yang and T. Mitchell, "Leveraging knowledge bases in lstms for improving machine reading," *arXiv preprint arXiv:1902.09091*, 2019.

[25] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *arXiv preprint arXiv:1610.02136*, 2016.

[26] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," *Advances in neural information processing systems*, vol. 31, 2018.

[27] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan, "Likelihood ratios for out-of-distribution detection," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[28] Y. Zheng, G. Chen, and M. Huang, "Out-of-domain detection for natural language understanding in dialog systems," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1198–1209, 2020.

[29] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril *et al.*, "Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces," *arXiv preprint arXiv:1805.10190*, 2018.

[30] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The atis spoken language systems pilot corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.

[31] B. Liu and I. Lane, "Recurrent neural network structured output prediction for spoken language understanding," in *Proc. NIPS Workshop on Machine Learning for Spoken Language Understanding and Interactions*, 2015.

[32] Y. Wu, Z. Zeng, K. He, H. Xu, Y. Yan, H. Jiang, and W. Xu, "Novel slot detection: A benchmark for discovering unknown slot types in the task-oriented dialogue system," *arXiv preprint arXiv:2105.14313*, 2021.

[33] G. Fei and B. Liu, "Breaking the closed world assumption in text classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 506–514.

[34] L. Shu, H. Xu, and B. Liu, "Doc: Deep open classification of text documents," *arXiv preprint arXiv:1709.08716*, 2017.

[35] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, 2009, pp. 147–155.