*Kỷ yếu Hội nghị Khoa học công nghệ Quốc gia lần thứ XVI về Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR); Đà Nẵng, ngày 28-29/9/2023*
*DOI: 10.15625/vap.2023.0087*

# NHẬN DIỆN BẢNG CHỮ CÁI NGÔN NGỮ KÝ HIỆU TIẾNG VIỆT SỬ DỤNG MÔ HÌNH HỌC SÂU

**Võ Đức Hoàng**

Khoa Công nghệ thông tin, Trường Đại học Bách Khoa, Đại học Đà Nẵng

*hoangvd.it@dut.udn.vn*

***TÓM TẮT:*** *Ngôn ngữ ký hiệu là phương tiện giao tiếp được sử dụng phổ biến trong cộng đồng người khiếm thính. Ngôn ngữ ký hiệu có những đặc trưng riêng với các quốc gia khác nhau, được biểu diễn thông qua các cử chỉ và hình dạng bàn tay, khuỷu tay, hay khuôn mặt. Việc nhận diện và giải mã ngôn ngữ ký hiệu là một thách thức lớn do sự phức tạp và đa dạng của các ký hiệu. Để giải quyết vấn đề này, các nhà nghiên cứu đã sử dụng mạng nơron sâu để trích xuất đặc trưng và phân loại ngôn ngữ ký hiệu. Các mô hình học sâu hiện nay đang tập trung nhiều về độ chính xác hơn kích thước của mô hình. Trong nghiên cứu đã đề xuất sử dụng và cải tiến mô hình SqueezeNet thử nghiệm trên bảng chữ cái ngôn ngữ ký hiệu tiếng Việt, cho độ chính xác 99.78% trên dữ liệu huấn luyện và nhận dạng bảng chữ cái tiếng Việt, cũng như số lượng tham số tốt hơn so với các phương pháp truyền thống. Mô hình này đóng góp vào việc ứng dụng cho các sản phẩm cuối đối với các thiết bị có cấu hình phần cứng hạn chế.*

***Từ khóa:*** *Mạng Squeeze-Net, Môđun fire, Mạng nơron tích chập, Ngôn ngữ ký hiệu tiếng Việt (VSL).*

## I. GIỚI THIỆU

Ngày nay, hệ thống thị giác máy tính được áp dụng nhiều trong các lĩnh vực như: giám sát, điều khiển công nghiệp, giao tiếp người và máy, truyền thông, điều khiển robot,... Có hai xu hướng nghiên cứu chính về nhận dạng ngôn ngữ ký hiệu tùy thuộc vào loại cử chỉ tĩnh hay động. Các nghiên cứu về nhận dạng ngôn ngữ ký hiệu tiếng Việt (Vietnamese Sign Language - VSL) tĩnh [1-5] đã cho các kết quả khá cao, ví dụ ở nghiên cứu nhận dạng VSL tĩnh được H.H. Hưng và cộng sự đưa ra vào năm 2012 [1], dữ liệu được thu nhận dưới dạng ảnh 2D thông qua camera màu. Sau khi trích xuất đặc trưng dựa trên hình dạng và đường bao, mạng nơron nhân tạo được sử dụng để thực hiện việc phân lớp. Việc thử nghiệm được thực hiện trên bộ ký hiệu tương ứng với bảng chữ cái tiếng Việt (nhiều ký tự hơn so với quốc tế) với độ chính xác lên đến 98%. Ngày nay, các mô hình học sâu được áp dụng rộng rãi trong các nghiên cứu nhận dạng. K.D. Bach và cộng sự [4] triển khai sử dụng mạng thần kinh tái phát (RNN) với khung theo dõi tay Mediapipe và sử dụng mô hình học sâu để nhận dạng cử chỉ ngôn ngữ ký hiệu cho kết quả tốt cho nhận dạng từng từ. Đặc biệt nghiên cứu gần đây nhất của Q.P. Van và cộng sự [5] đề xuất một phương pháp mới để nhận dạng VSL (Ngôn ngữ ký hiệu tiếng Việt) dựa trên sự kết hợp giữa phân bố Gauss và hệ số tương quan để trích xuất đối tượng động duy nhất trong video là GoogLeNet (mô hình CNN) và BiL-STM (Bidirectional Long Short-Term). Memory) để phân loại chuỗi video. Bộ dữ liệu được sử dụng bao gồm 2700 mẫu tương ứng với 27 dấu hiệu trong VSL. Kết quả thử nghiệm đạt độ chính xác 99,38% với bộ xác thực và 98,15% với bộ kiểm tra.

Ngôn ngữ ký hiệu có những đặc điểm đặc biệt khiến cho việc nhận dạng trở nên phức tạp. Việc biểu thị ý nghĩa bằng các hình dáng, vị trí, hướng và độ mở của tay tạo ra nhiều biến thể khác nhau. Ngay cả khi cùng một người sử dụng ngôn ngữ ký hiệu, ký hiệu của họ tại hai thời điểm khác nhau có thể khác nhau hoàn toàn. Những yếu tố khách quan như sức khỏe và tâm trạng cũng có thể ảnh hưởng đến độ mở và vị trí của tay, làm cho quá trình nhận dạng trở nên khó khăn hơn. Vì vậy, một mô hình nhận dạng hiệu quả phải có khả năng trích chọn và học những đặc trưng của các ngôn ngữ ký hiệu.

Các nghiên cứu gần đây về các mạng học sâu sử dụng nơron tích chập (CNN) thường tập trung vào việc tăng độ chính xác trên các bộ dữ liệu thị giác máy tính. Với một độ chính xác nhất định, thường có nhiều kiến trúc CNN khác nhau đạt được độ chính xác đó. Với độ chính xác tương đương, một kiến trúc CNN có số lượng tham số ít hơn có nhiều lợi ích:

- Tăng hiệu quả huấn luyện phân tán. Giao tiếp giữa các máy chủ là yếu tố giới hạn đến tính mở rộng của việc đào tạo CNN phân tán. Đối với việc đào tạo phân tán dữ liệu song song, chi phí giao tiếp tỷ lệ thuận trực tiếp với số lượng tham số trong mô hình [6]. Tóm lại, các mô hình nhỏ đào tạo nhanh hơn do yêu cầu ít giao tiếp hơn.
- Giảm chi phí khi khi cập nhật mô hình mới cho khách hàng. Đối với xe tự hành, các công ty như Tesla thường sao chép các mô hình mới từ máy chủ của họ vào ô tô của khách hàng. Quá trình này thường được gọi là cập nhật OTA (over-the-air). Các báo cáo khách hàng đã phát hiện ra rằng tính an toàn của chức năng lái xe bán tự động của Tesla đã được cải thiện theo từng bước với các cập nhật OTA gần đây (báo cáo khách hàng, 2016 [7]). Tuy nhiên, các cập nhật OTA của các mô hình CNN/DNN hiện nay có thể yêu cầu việc truyền tải dữ liệu lớn. Với mô hình AlexNet, việc này cần phải tải xuống 240 MB dữ liệu từ máy chủ đến ô tô. Các mô hình nhỏ hơn đòi hỏi ít giao tiếp hơn, giúp cho các bản cập nhật thường xuyên trở nên khả thi hơn.
- Triển khai trên các thiết bị FPGA và nhúng khả thi hơn. FPGA thường có ít hơn 10 MB bộ nhớ trên chip và không có bộ nhớ hoặc lưu trữ ngoài chip. Đối với việc sử dụng mô hình, một mô hình đủ nhỏ có thể được lưu trữ trực tiếp trên FPGA thay vì bị thắt cổ chai bởi băng thông bộ nhớ [8], trong khi các khung hình video được truyền qua FPGA trong thời gian thực. Hơn nữa, khi triển khai CNN trên mạch tích hợp ứng dụng cụ thể

(ASIC), một mô hình đủ nhỏ có thể được lưu trữ trực tiếp trên chip và các mô hình nhỏ hơn có thể giúp cho ASIC vừa với một mảnh chip nhỏ hơn.

Với những ưu điểm này của một kiến trúc CNN có ít tham số, bài báo này tập trung trực tiếp vào vấn đề xác định một kiến trúc CNN có số lượng tham số ít hơn nhưng độ chính xác tương đương so với các mô hình truyền thống khác. Bài báo đã sử dụng mô hình SqueezeNet [9] đối với bài toán nhận diện ngôn ngữ ký hiệu tiếng Việt và ngoài ra, bài báo cũng trình bày các so sánh, đánh giá so với các mô hình nổi tiếng khác. Mục đích của việc so sánh này nhằm đưa ra một bức tranh tổng thể về các kỹ thuật học sâu được sử dụng cho việc nhận dạng ngôn ngữ ký hiệu tiếng Việt.

## II. XÂY DỰNG MÔ HÌNH NHẬN DIỆN VSL VỚI SQUEEZENET

Phần này sẽ trình bày về cách xây dựng module Fire, thành phần chính trong mạng SqueezeNet và cách xây dựng mô hình SqueezeNet để nhận diện ngôn ngữ ký hiệu tiếng Việt.

### *A. Thiết kế kiến trúc*

Mục tiêu của nghiên cứu này là tìm ra kiến trúc CNN có ít tham số nhưng vẫn đảm bảo được độ chính xác. Để đạt được điều đó, mô hình CNN sẽ tuân theo 3 chiến lược sau:

*1.* Thay thế các bộ lọc 3x3 thành 1x1

Với một số lượng lớp tích chập nhất định, mô hình sẽ chọn phần lớn các bộ lọc có kích thước 1x1, vì một bộ lọc 1x1 có số lượng tham số ít hơn 9 lần so với bộ lọc 3x3.

*2.* Giảm số lượng đầu vào cho các bộ lọc 3x3

Xét một lớp tích chập được tạo thành hoàn toàn từ các bộ lọc 3x3. Tổng số lượng tham số trong lớp này là (số kênh đầu vào) * (số bộ lọc) * (3*3). Vì vậy, để giữ cho tổng số lượng tham số nhỏ trong một mô hình CNN, chúng ta không chỉ cần phải giảm số lượng bộ lọc 3x3 (như chiến lược 1 ở trên), mà còn cần phải giảm số lượng đầu vào cho các bộ lọc 3x3. Tác giả giảm số lượng kênh đầu vào cho các bộ lọc 3x3 bằng cách sử dụng các lớp squeeze, mà bài báo mô tả trong phần tiếp theo.

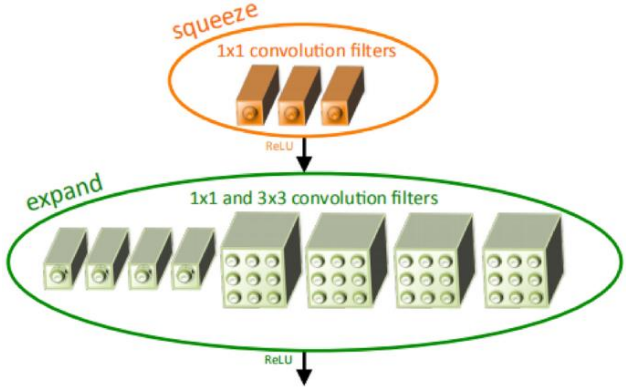
*3.* Giảm kích thước mẫu trong mạng để các lớp tích chập có bản đồ kích hoạt lớn

Trong một mạng tích chập, mỗi lớp tích chập tạo ra một bản đồ kích hoạt đầu ra có độ phân giải không nhỏ hơn 1x1 và thường lớn hơn nhiều so với 1x1. Chiều cao và chiều rộng của các bản đồ kích hoạt này được xác định bởi: kích thước của dữ liệu đầu vào và sự lựa chọn các lớp để giảm mẫu trong kiến trúc CNN. Thông thường, việc giảm mẫu được tích hợp vào kiến trúc CNN bằng cách thiết lập cửa sổ trượt lớn hơn 1 trong một số lớp tích chập hoặc pooling [5, 6, 7]. Nếu các lớp đầu trong mạng có cửa sổ trượt lớn, thì hầu hết các lớp sẽ có bản đồ kích hoạt nhỏ. Ngược lại, nếu hầu hết các lớp trong mạng có cửa sổ trượt bằng 1 và các cửa sổ trượt lớn hơn 1 được tập trung vào cuối mạng, thì nhiều lớp trong mạng sẽ có bản đồ kích hoạt lớn. Theo quan sát, bản đồ kích hoạt lớn (do giảm mẫu chậm) có thể dẫn đến độ chính xác phân loại cao hơn, với tất cả các yếu tố còn lại được giữ nguyên. [9]

Chiến lược 1 và 2 nhằm giảm số lượng tham số trong một mạng tích chập nhưng vẫn cố gắng giữ nguyên độ chính xác. Chiến lược 3 nhằm tối đa hóa độ chính xác với một số lượng lớp tích chập hạn chế.

### *B. Xây dựng môđun Fire*

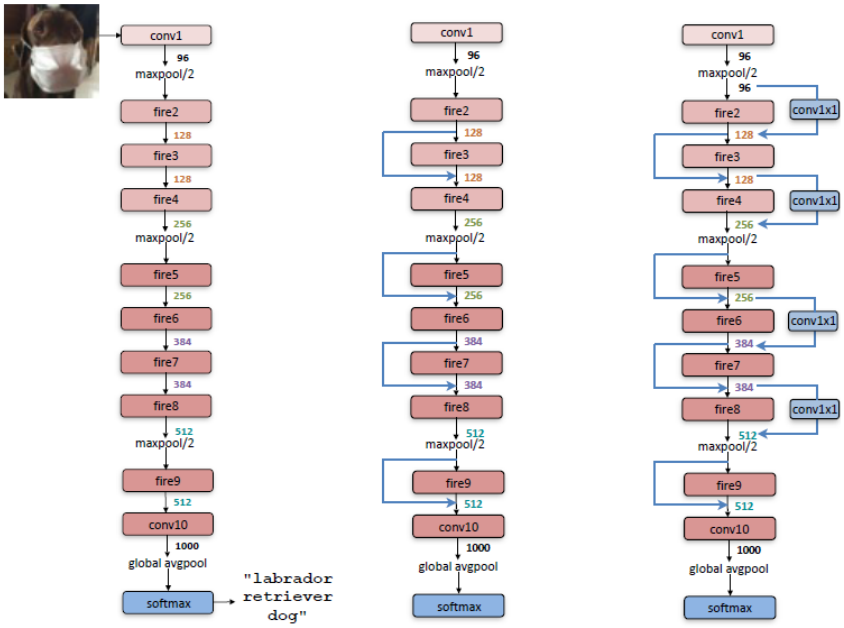
Môđun Fire được định nghĩa như sau: Một Fire môđun bao gồm: một lớp tích chập *squeeze* (chỉ có 1 bộ lọc 1x1), đưa vào một lớp *expand*, là sự kết hợp của bộ lọc tích chập 1x1 và 3x3; minh họa trong Hình 1. Module Fire có 3 siêu tham số có thể điều chỉnh: $s_{1x1}$, $e_{1x1}$ và $e_{3x3}$. Trong một module Fire, $s_{1x1}$ là số bộ lọc trong lớp squeeze (tất cả đều có kích thước 1x1), $e_{1x1}$ là số bộ lọc 1x1 trong lớp mở rộng và $e_{3x3}$ là số bộ lọc 3x3 trong lớp mở rộng. Tham số $s_{1x1}$ được khai báo nhỏ hơn ($e_{1x1}$ + $e_{3x3}$), vì vậy lớp squeeze giúp giới hạn số kênh đầu vào cho các bộ lọc 3x3.




**Hình 1.** Môđun Fire với tham số $s_{1x1}$=3, $e_{1x1}$=4, $e_{3x3}$=4

### *C. Xây dựng mô hình SqueezeNet*

Kiến trúc của mô hình SqueezeNet CNN được minh hoạ như Hình 2. SqueezeNet bắt đầu với một lớp tích chập độc lập (conv1), tiếp theo là 8 lớp module Fire (Fire2-9) và kết thúc bằng một lớp tích chập cuối cùng (conv10). Số bộ lọc của mỗi fire module được tăng dần từ đầu đến cuối mạng. SqueezeNet thực hiện max-pooling với cửa sổ trượt là 2 sau các lớp conv1, fire4, fire8 và conv10; các vị trí pooling tương đối muộn này giúp cho các lớp tích chập có bản đồ kích hoạt lớn hơn.

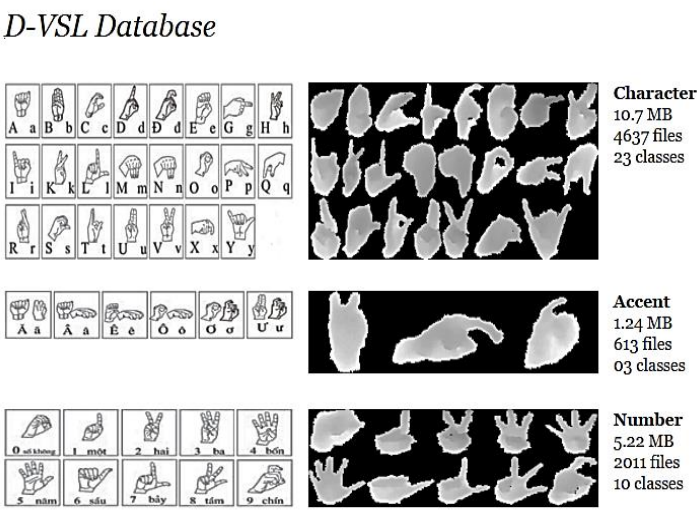



**Hình 2.** Kiến trúc của mô hình SqueezeNet

## III. THỰC NGHIỆM VÀ KẾT QUẢ

### *A. Dữ liệu*

Tập dữ liệu bảng chữ cái ngôn ngữ ký hiệu tiếng Việt (có thể tải xuống tại địa chỉ http://test101.udn.vn/d-vsl/). Bộ dữ liệu được chia thành 3 phần riêng biệt. Phần chữ bao gồm 23 ký tự đơn của bảng chữ cái là gồm các ký tự đơn giản, không bao gồm dấu. Phần mũ bao gồm dấu mũ kết hợp để tạo ra các chữ cái (đặc trưng của ngôn ngữ ký hiệu tiếng Việt): ă, â, ê, ô, ơ ư. Phần số là các mẫu ký hiệu các số từ 0 đến 9. Dữ liệu được chia sẵn thành tập huấn luyện và tập kiểm tra. Trong mỗi tập con có các thư mục, mỗi thư mục là 1 chữ cái, dấu, số và 1 thư mục cho ảnh ngẫu nhiên không chứa ký hiệu. Tỉ lệ bộ dữ liệu huấn luyện và kiểm tra 9:1 tương đương ví dụ ở Bảng chữ cái là 4174 tập huấn luyện và 463 tập xác thực. Tiếp theo, lấy trong tập huấn luyện 20% mẫu cho tập xác thực (validate). Lúc tập xác thực có 4174 mẫu và tập kiểm tra có 1298 mẫu.

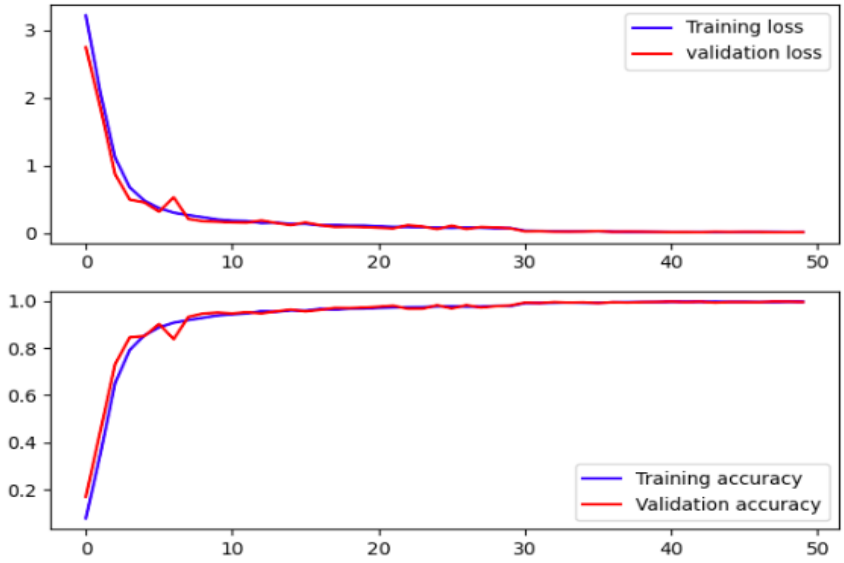



**Hình 3.** Bộ dữ liệu hình ảnh độ sâu của NNKH tiếng Việt

### *B. Tiền xử lý dữ liệu*

Ảnh được điều chỉnh về kích thước (224 x 224 pixels) để phù hợp với đầu vào của mô hình. Sau đó giá trị của các điểm ảnh sẽ được chia cho 255 để được chuẩn hoá về khoảng [0-1] (ảnh được chuẩn hoá sẽ giúp mô hình hội tụ nhanh hơn, giúp quá trình huấn luyện hiệu quả hơn).

### *C. Quá trình huấn luyện*




**Hình 4.** Biểu đồ minh hoạ độ chính xác và hàm mất mát của tập huấn luyện và tập xác thực trong quá trình huấn luyện mô hình SqueezeNet

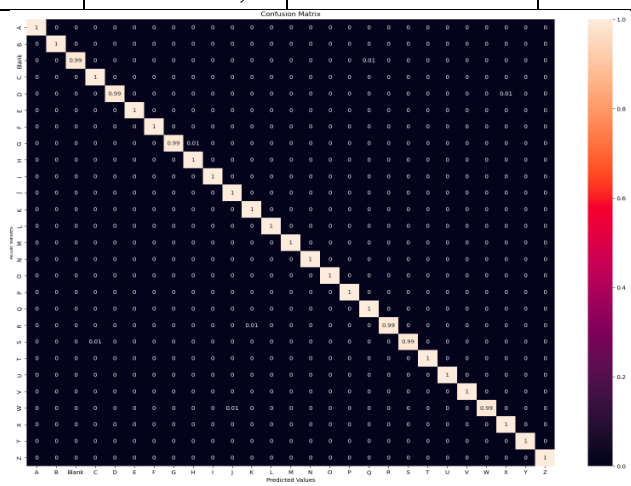
Mô hình được huấn luyện dựa trên phương pháp Gradient Descent để cập nhật trọng số cho đến khi hội tụ. Tốc độ học (learning rate) ban đầu là 0,01 và được giảm dần theo cấp số nhân 0,3 đến tối thiểu $10^{-7}$. Tốc độ học sẽ giảm khi độ chính xác của tập xác thực không tăng trong 3 epoch liên tiếp. Quá trình học sẽ bị buộc dừng khi độ chính xác của tập xác thực không tăng trong 10 epoch liên tiếp, khi đó mô hình sẽ khôi phục lại trọng số của epoch có kết quả tốt nhất.

### *D. Kết quả*

Trong mục này, bài báo so sánh độ chính xác, số lượng tham số và dung lượng của mô hình SqueezeNet với các mô hình nổi tiếng khác. Với mỗi mô hình, ta thay đổi các thông số để có được kết quả tốt nhất. Chú ý rằng các mô hình này có sự khác biệt liên quan đến kiến trúc, cũng như độ phức tạp tính toán. Kết quả được thể hiện trong Bảng 1.

**Bảng 1.** Kết quả của các mô hình áp dụng cho ký tự (character)

| Mô hình | Độ chính xác | Số lượng tham số | **Dung lượng** |
|---|---|---|---|
| **SqueezeNet** | **99,78** | **666.555** | ~8.4 MB |
| VGG16 | 97,26 | 17.966.043 | ~98 MB |
| VGG19 | 95,41 | 23.275.739 | ~119.2 MB |
| Xception | 89,41 | 33.746.627 | ~238.4 MB |




**Hình 5.** Ma trận nhiễu của mô hình SqueezeNet đối với tập kiểm tra

Dựa trên kết quả trong Bảng 1, ta thấy rằng mô hình SqueezeNet cho kết quả tốt nhất so với các mô hình khác (VGG16, VGG19, Xception). Trong đó, mô hình SqueezeNet có độ chính xác cao, với 99,78%, số lượng tham số ít nhất, chỉ 666,555 tham số và chiếm ít dung lượng bộ nhớ nhất, chỉ xấp xỉ 8,4 MB cho toàn bộ dữ liệu các ký hiệu bảng chữ cái. Bằng cách sử dụng các lớp tích chập 1x1 thay vì 3x3 như truyền thống, số lượng tham số của mô hình đã giảm đi đáng kể nhưng độ chính xác vẫn không suy giảm. So nghiên cứu của P.T. Hai năm 2018 [14] sử dụng máy vectơ hỗ trợ đa lớp (SVMs) và One-Against-All (OAA) kết quả nhận dạng đạt 90,29% cho 15 từ huấn luyện và nhận dạng. Năm 2021, K.D. Bach và cộng sự đã sử dụng phương pháp Mediapipe để nhận dạng hình ảnh bàn tay với 15 cử chỉ tay, tuy kết quả đạt được khả quan nhưng vẫn còn ở mức thấp khoảng 80%.

Ma trận trong Hình 5 cho thấy mô hình thể hiện rất tốt, mô hình có rất ít trường hợp bị nhầm lẫn giữa các lớp. Tuy nhiên, vẫn có những hình ảnh khó với chất lượng, độ sáng không ổn định khiến mô hình nhận diện sai.

 G  H

**Hình 6.** Hình ảnh thực tế của chữ cái G và H trong bảng chữ cái ngôn ngữ ký hiệu tiếng Việt


**Hình 7.** Hình ảnh thực tế nhận dạng của chữ cái G thành chữ H

## IV. KẾT LUẬN

Bài báo này đã trình bày bài toán nhận diện ngôn ngữ ký hiệu với số lượng tham số tối giản, đây là vấn đề có ý nghĩa quan trọng đối với việc nhúng mô hình vào các thiết bị cuối. Đặc biệt hướng đến xây dựng các ứng dụng học tập ngôn ngữ ký hiệu trên thiết bị di động cho người khiếm thính. Nghiên cứu đã xem xét nhiều mô hình học sâu khác nhau như VGG16, VGG19, Xception. Kết quả thực nghiệm cho thấy, mạng SqueezeNet cho ra độ chính xác cao nhất cũng như chiếm ít dung lượng bộ nhớ nhất. Với kết quả nhận dạng với độ chính xác cao trên bộ dữ liệu do chính tác giả xây dựng là tiền đề cho các nghiên cứu tiếp theo trong việc nhận dạng các từ và các hình ảnh ngôn ngữ ký hiệu được tách từ các video.

## TÀI LIỆU THAM KHẢO

[1] Hưng, Huỳnh Hữu, ccs., "Nhận dạng ngôn ngữ ký hiệu tiếng Việt sử dụng mạng Neuron nhân tạo," *Tạp chí Khoa học và Công nghệ, Đại học Đà Nẵng*, số 12.61: 75-80, 2012.

[2] T. N. Thi Huong, T. V. Huu, T. L. Xuan and S. V. Van, "Static hand gesture recognition for vietnamese sign language (VSL) using principle components analysis," *International Conference on Communications, Management and Telecommunications (ComManTel)*, DaNang, Vietnam, 2015, pp. 138-141, DOI: 10.1109/ComManTel.2015.7394275.

[3] V. D. Nguyen, M. T. Chew and S. Demidenko, "Vietnamese sign language reader using Intel Creative Senz3D," *6th International Conference on Automation, Robotics and Applications (ICARA)*, Queenstown, New Zealand, 2015, pp. 77-82, DOI: 10.1109/ICARA.2015.7081128.

[4] Duy Khuat, Bach, et al. "Vietnamese sign language detection using Mediapipe," *10th International Conference on Software and Computer Applications*, 2021.

[5] Q. P. Van and B. N. Thanh, "Vietnamese Sign Language Recognition using Dynamic Object Extraction and Deep Learning," *2020 IEEE Eighth International Conference on Communications and Electronics (ICCE)*, Phu Quoc Island, Vietnam, 2021, pp. 402-407, DOI: 10.1109/ICCE48956.2021.9352071.

[6] Forrest N. Iandola, Khalid Ashraf, Matthew W. Moskewicz, and Kurt Keutzer, "FireCaffe: near-linear acceleration of deep neural network training on compute clusters," *CVPR*, 2016.

[7] "Consumer Reports. Tesla's new autopilot: Better but still needs improvement," http://www.consumerreports.org/tesla/tesla-new-autopilot-better-but-needs-improvement, 2016.

[8] Jiantao Qiu, Jie Wang, Song Yao, Kaiyuan Guo, Boxun Li, Erjin Zhou, Jincheng Yu, Tianqi Tang, Ningyi Xu, Sen Song, Yu Wang, and Huazhong Yang, "Going deeper with embedded fpga platform for convolutional neural network," *ACM International Symposium on FPGA*, 2016.

[9] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, Kurt Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size," *arXiv: 1602.07360*, 2016.

[10] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, "Going deeper with convolutions," *arXiv: 1409.4842*, 2014.

[11] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv: 1409.1556*, 2014.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *NIPS*, 2012.

[13] Kaiming He and Jian Sun, "Convolutional neural networks at constrained time cost," *CVPR*, 2015.

[14] Hai, Pham The, et al., "Automatic feature extraction for Vietnamese sign language recognition using support vector machine," *2018 2nd International Conference on Recent Advances in Signal Processing, Telecommunications & Computing (SigTelCom)*. IEEE, 2018.

# RECOGNITION OF VIETNAMESE SIGNAL LANGUAGE USING DEEP LEARNING METHODS

**Vo Duc Hoang**

***ABSTRACT:*** *Sign language is a commonly used means of communication in the deaf community. Sign languages have unique characteristics for different countries, expressed through gestures and hand, elbow, or facial shapes. Recognizing and decoding sign language is a major challenge due to the complexity and diversity of symbols. To solve this problem, researchers used deep neural networks to extract features and classify sign languages. Current deep learning models are focusing more on accuracy than model size. In the study, it was proposed to use and improve the SqueezeNet model tested on the Vietnamese sign language alphabet, giving 99.78% accuracy on training data and recognizing the Vietnamese alphabet, as well as numbers. The number of parameters is better than traditional methods[1]. This model contributes to end-product applications for devices with limited hardware configurations.*