



AI BUILDERS

# Dog Breed Classification

Pakin Olanraktham

## Abstract

โครงการนี้ทำเพื่อพัฒนา **Machine Learning** สำหรับจำแนกสายพันธุ์สุนัข โดยใช้ **ResNet50** เทรนไป 10 Epoch ด้วย **Fast AI** ซึ่งผลลัพธ์ที่ได้ พบว่ามีความแม่นยำถึง **87%**

## Introduction & Motivation

คุณเคยเจอปัญหาเหล่านี้หรือไม่ เจอสุนัขอยู่ข้างทาง อยากรู้ว่ามันเป็นสายพันธุ์อะไร แต่เชิร์ชหาไม่เจอ ปัญหาเหล่านี้จะหมดไป เพราะจะมี Machine Learning เข้ามาช่วย!

ในปัจจุบันมีสุนัขอยู่ประมาณ 190 สายพันธุ์! ซึ่งเราคงจำหน้าตามันไม่ได้ทั้งหมดและบางสายพันธุ์อาจมีนิสัย, ความฉลาด, ความดุ, ฯลฯ ที่แตกต่างกัน ถ้าเรารู้สายพันธุ์ของสุนัขแต่ละตัว เราสามารถไปค้นหาข้อมูลเพิ่มเติมได้ และเราก็จะรู้จักกับสุนัขสายพันธุ์นั้นมากขึ้น ดังนั้นการที่จะนำ **Machine Learning** เข้ามาช่วยแยกสายพันธุ์สุนัขน่าจะเป็นสิ่งที่มีประโยชน์

## Data Sets

หลังจากทำการหา Data Set และเจอ Data Set ชื่อ “**Stanford Dogs Dataset**”<sup>[1]</sup> ซึ่งมีสุนัขทั้งหมด 120 สายพันธุ์ รวม 20580 รูป (เฉลี่ยสายพันธุ์ ละ 171.5 รูป) แต่พอ Download Dataset มาจะได้เป็นไฟล์ที่ยังไม่ได้แยก Train กับ Test จึงไปหาข้อมูลเพิ่มเติมแล้วไปเจอ Dataset นี้อยู่ใน website ของ **TensorFlow**<sup>[2]</sup> ซึ่งมี Train กับ Test แยกมาให้ ซึ่งแต่ละสายพันธุ์จะมีรูปอยู่ใน Train set สายพันธุ์ละ 100 รูปเท่ากัน

ถ้าเกิดเราหารูปเพิ่มแล้วใส่ลงใน Train set, model อาจมีความแม่นยำมากขึ้นหรือไม่ จึงได้ทำการสร้าง Data set อีก version แล้ว Scrape รูปมาใส่ใน Folder ของ Train set (ได้รูปภาพเฉลี่ย Class ละ 370 รูป), แต่การ Scrape รูปจาก DuckDuckGo อาจได้รูปที่ไม่ถูกต้องเช่นการใช้ keyword เป็น “Siberian Husky” อาจได้รูปของ “เสื้อลาย Siberian Husky” ก็เลยต้องลบด้วยมือ แต่ทำแค่นี้ยังไม่พอ เพราะอาจมีรูปพวกนี้ไปซ้ำอยู่ใน Test Set! จึงได้ลองใช้ Library ที่มีชื่อ “**diffPy**”<sup>[3]</sup> ซึ่งเป็น Library ที่สามารถหารูปที่ซ้ำกันได้, หลังจากลองประมวลผลไป 1 ครั้ง โปรแกรมก็ลบรูปที่ซ้ำกันไป 1 รูปได้ตามปกติ แต่ว่าอาจจะมียังหาไม่เจอเลยต้องรันไป 4-5 ครั้ง เพื่อที่จะได้แน่ใจว่าไม่มีรูปซ้ำกัน

## Methods

**เริ่มเทรนด้วยวิธีที่ 1 — ใช้ Fast AI เทรนด้วย Dataset ที่ยังไม่ได้ Scrape รูปมาเพิ่ม**

โดยการเทรนจะลองใช้ ResNet50, VGG16 และ VGG19 แล้วนำทั้งสามโมเดลมาเปรียบเทียบกับพอเทรนโมเดลเสร็จ จึงค้นพบว่า โมเดลที่ได้ Accuracy สูงที่สุดคือ **ResNet50** ซึ่งได้ Accuracy อยู่ที่ **84.75%** เมื่อทดสอบกับ Validation Set, เลยเลือกที่จะใช้ ResNet50 ในการเทรนโมเดลด้วยวิธีอื่นๆ, พอลองดู Most Confuse จะเห็นว่าเป็น 097\_Eskimo\_Dog กับ 099\_Siberian\_Husky Confuse กันเยอะที่สุด ซึ่งทั้งสองสายพันธุ์นี้มีหน้าตาที่คล้ายกัน!<sup>[4]</sup> จึงตั้งสมมติฐานว่าข้อผิดพลาดส่วนใหญ่ของโมเดลนี้ เกิดจากแต่ละสายพันธุ์มีลักษณะหน้าตาที่คล้ายกัน



**เริ่มเทรนโมเดลด้วยวิธีที่ 2 — ใช้ Fast AI เทรนด้วย Dataset ที่ไป Scrape รูปมาเพิ่มแล้ว**

หลังจากเทรนไป 10 Epoch ด้วย ResNet50 เหมือนโมเดลที่แล้ว แต่เปลี่ยน Data Set เป็น Data Set ที่ไป Scrape รูปมาเพิ่ม พอเทรนเสร็จแล้ว โมเดลได้ Accuracy อยู่ที่ **87.33%** เมื่อทดสอบกับ Validation Set, ต่อมาเลยลองมาดู Most Confuse ของโมเดลนี้ก็พบว่า Most Confuse คือ 080\_Collie กับ 081\_Border Collie ซึ่งก็มีลักษณะคล้ายกันอีกแล้ว<sup>[5]</sup>, จึงตั้งสมมติฐานว่าข้อผิดพลาดส่วนใหญ่เป็นเพราะสุนัขแต่ละสายพันธุ์มีลักษณะคล้ายกันเหมือนครั้งก่อนที่เทรนไป



## ทดสอบ 2 โมเดลแรกด้วย Test Set

เนื่องจากทั้งสองโมเดลไม่ได้ถูกทดสอบด้วยชุดของรูปภาพที่เหมือนกัน เลยต้องลองทดสอบด้วย Test Set แล้วได้ผลว่า โมเดลแรกได้ Accuracy อยู่ที่ **86%** และ โมเดลที่สองได้ Accuracy อยู่ที่ **87%**

พอลองมาดู Confusion Matrix แล้วสังเกตดีๆ จะเห็นว่ามีบางคลาสที่โมเดลใหม่ตอบถูกเยอะกว่าโมเดลเก่าและมีบางคลาสที่โมเดลเก่าตอบถูกเยอะกว่าโมเดลใหม่ แต่โดยรวมถือว่าโมเดลใหม่มีความแม่นยำกว่าโมเดลเก่า

**เริ่มเทรนด้วยวิธีที่ 3 — ใช้ Pytorch เทรนด้วย Dataset ที่ไป Scrape รูปมาเพิ่มแล้ว**

โดยการเขียนด้วย Pytorch จะมีความซับซ้อนกว่าการเขียนด้วย Fast AI แต่มีวิธีที่คล้ายๆกัน

หลังจากเทรนไป 10 Epoch ด้วย ResNet50, โมเดลได้ Accuracy แค **67%** เมื่อทดสอบด้วย Test Set ซึ่งน้อยกว่าโมเดลที่ถูกเทรนด้วย Fast AI มาก, จึงตั้งสมมติฐานว่าเป็นเพราะ Fast AI จำกำหนดมาให้ แต่ Pytorch สามารถกำหนด Augmentation ได้เอง ซึ่งอาจจะทำให้รูปเพี้ยนจากความเป็นจริงเกินไป และอาจจะเป็นเพราะไม่ได้หา Learning Rate ที่เหมาะสมเหมือนกับ Fast AI

## Results

ได้โมเดลที่ดีที่สุดคือโมเดลที่เทรนด้วย **Fast AI** กับ Dataset ที่ไป Scrape รูปมาเพิ่ม ซึ่งได้ Accuracy กับ Test Set อยู่ที่ **87%** ซึ่งข้อผิดพลาดส่วนใหญ่เกิดจากการที่สายพันธุ์ของสุนัขมีลักษณะหน้าตาที่คล้ายกัน

## Deployment

การทำ Deployment จะใช้ Streamlit เพราะเป็น open-source app framework ที่ใช้แค่ความรู้ด้าน Python ก็สามารถทำ website ได้โดยไม่ต้องใช้ html หรือ css

โดยตัวแอฟจะสามารถอัปโหลดรูปเพื่อไปทำนายสายพันธุ์ของสุนัขได้หรือจะเลือกรูปภาพจาก Test Set ได้เลยก็ได้ ซึ่งพอทำนายออกมาแล้วจะบอกความน่าจะเป็นที่จะตอบถูกและสายพันธุ์ที่ทำนายได้

## Future Development

1. Clean Data ให้ดีขึ้นเพราะอาจมีรูปสุนัขที่ไปอยู่ผิดคลาสเนื่องจากมันหน้าตาคล้ายกัน
2. ทำให้โมเดลสามารถแยกได้ดียิ่งกว่ามันเป็นหรือไม่เป็นสุนัข เช่น การใส่รูปสัตว์ต่างๆ ที่ไม่ใช่สุนัขเพื่อไปทำนาย จะได้ผลลัพธ์ว่าเป็นสายพันธุ์ใดสายพันธุ์หนึ่งของสุนัขอยู่ดี ไม่ได้บอกว่าไม่ใช่สุนัข
3. เพิ่มจำนวนสายพันธุ์สุนัข เพราะว่ามีสุนัขบางสายพันธุ์ที่เรารู้จัก เช่น บางแก้ว แต่ไม่ได้ถูกเทรน พอนำไปทำนายผล ก็จะตอบเป็นสายพันธุ์อื่น
4. ทำ Fine-grained Classification เนื่องจากบางสายพันธุ์มีลักษณะที่คล้ายกัน การเพิ่ม Data หรือเทรนโมเดลเพิ่ม อาจจะได้ช่วยให้ Accuracy ดีขึ้น แต่การทำ Fine-grained Classification<sup>[6]</sup> สามารถแยกสิ่งของที่มีลักษณะคล้ายๆ กันได้ดีกว่าการเทรนแบบปกติ

## References

1. <http://vision.stanford.edu/aditya86/ImageNetDogs/>
2. [https://www.tensorflow.org/datasets/catalog/stanford\\_dogs](https://www.tensorflow.org/datasets/catalog/stanford_dogs)
3. <https://github.com/elisemercury/Duplicate-Image-Finder>
4. [https://images.saymedia-content.com/.image/t\\_share/MTc2Mjg3NTg2MDM2MzYwMzY1/11-dog-breeds-like-husky.png](https://images.saymedia-content.com/.image/t_share/MTc2Mjg3NTg2MDM2MzYwMzY1/11-dog-breeds-like-husky.png)
5. <https://a-z-animals.com/media/2022/03/c8f0830a519ed6ec2faa6d9ffb50eee481ed17d5-1024x614.jpg>
6. <https://paperswithcode.com/task/fine-grained-image-classification>

