

Metodologi *Predictive Analysis* Umur berdasarkan *Image Big Data* *Challenge Satria Data 2021*

Widyanto Hadi Nugroho
Fakultas Ilmu Komputer
Universitas Indonesia
Depok, Jawa Barat
widyanto@ristek.cs.ui.ac.id

Geoffrey Tyndall
Fakultas Ilmu Komputer
Universitas Indonesia
Depok, Jawa Barat
geoffrey.tyndall@ristek.cs.ui.ac.id

Ardhani Dzaky Ralfiano
Fakultas Ilmu Komputer
Universitas Indonesia
Depok, Jawa Barat
ardhanidzaky@ristek.cs.ui.ac.id

A. Dataset Exploration

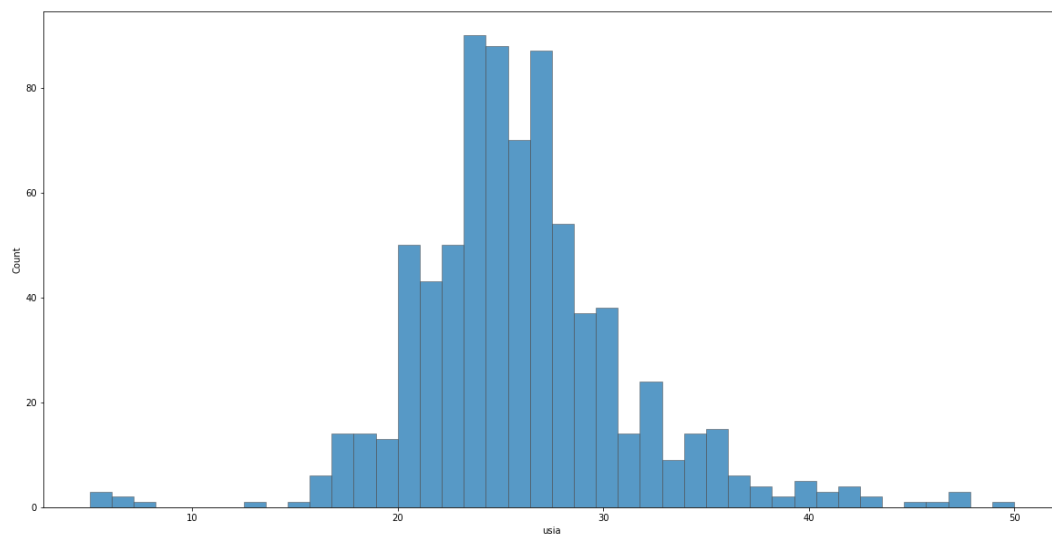
Dataset yang diberikan merupakan data dalam bentuk *image*. Data telah disediakan oleh pihak panitia Satria Data 2021 sejumlah 2.310 *image* untuk *training* dan 990 *image* untuk *testing*.

Foto yang digunakan dalam lomba BDC 2021 ini adalah foto yang berasal dari ruang publik internet terdiri atas 3 foto untuk setiap individu. Satu foto dapat merupakan foto individu orang tersebut, atau juga foto bersama.

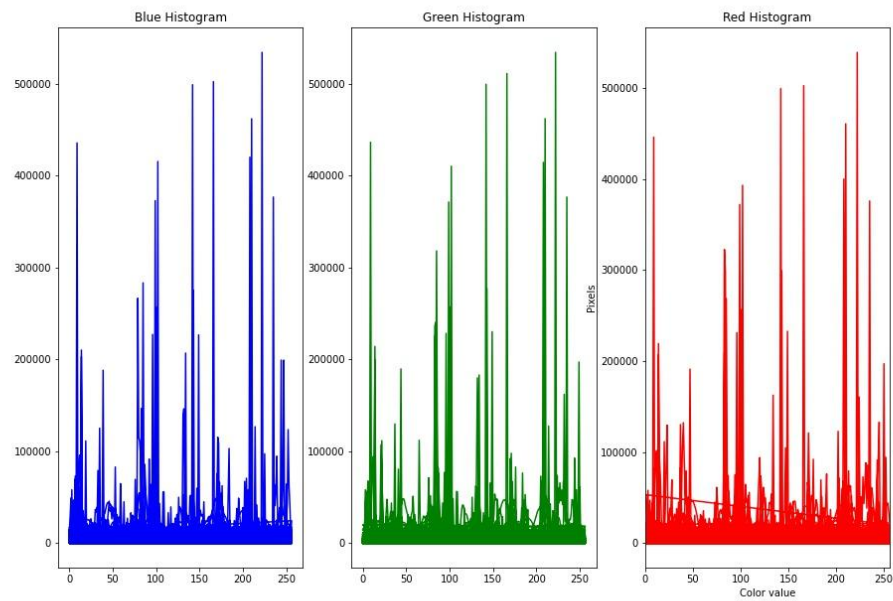
Berikut adalah hasil eksplorasi dataset.

1. Persebaran Umur

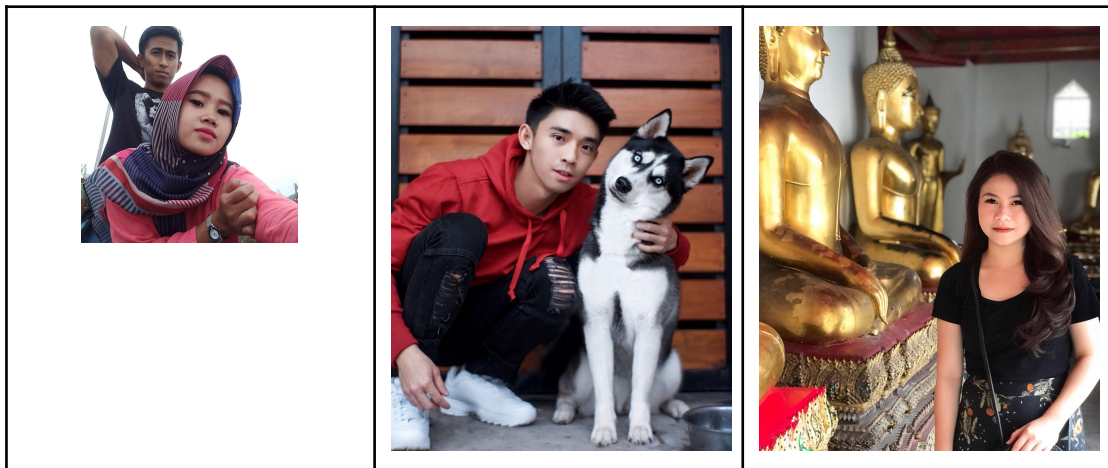
Persebaran umur yang terdapat pada data *training* kurang merata dengan umur 20 sampai 30 tahun mendominasi jumlah data.



2. *Image* dalam dataset memiliki format berwarna atau RGB (*Red, Green, Blue*). Analisis histogram nilai RGB dalam dataset menunjukkan variansi yang tinggi.




3. *Image* dalam dataset merupakan hasil pengambilan potret subjek yang ingin dideteksi. Namun, banyak *image* mengandung *noise objects*, seperti: subjek manusia lain, subjek binatang, objek wisata, dsb. Berikut beberapa di antaranya.



B. Preprocessing

1. *Cropping image*

Dataset *training* maupun *testing* melalui proses *cropping image* terlebih dahulu sehingga proses training memakai *dataset* dengan bentuk *full face*. *Cropping image* pada bagian wajah menggunakan bantuan *library autocrop*. Parameter yang digunakan pada Cropper adalah *face percentage* (100). Hal ini akan menghasilkan image dengan data train berupa wajah langsung dan menghindari *noise* dari *object* selain wajah pada gambar. Hal ini terbukti sangat efektif karena setelah melakukan hal ini, model kami memiliki performa yang jauh lebih baik.

Before Cropping	After Cropping
	

2. *Augmenting*

Sebelum data masuk ke dalam proses *training*, kami melakukan proses augmentasi data menggunakan beberapa cara yaitu,



- *IntToTensor()*

Kami mengubah angka-angka pixel pada image menjadi tensor agar dapat digunakan pada GPU.

- *Resize(300, ResizeMethod.Squish)*



Karena *image* hasil dari *cropping* yang kami lakukan memiliki dimensi yang berbeda-beda, kami melakukan metode *resize*. Hal tersebut kami lakukan agar *image* yang akan dimasukkan sebagai fitur pada model memiliki dimensi yang sama. Dengan dimensi yang sama, algoritma model dapat berjalan memperoleh jenis informasi yang sama dari setiap *image*.

Method *resize* yang kami gunakan adalah *squish* yang akan melakukan pelebaran dari *image* asli hingga memenuhi dimensi yang diinginkan sehingga *image* menjadi terlihat seperti tertarik atau *flattened*. Metode ini memungkinkan proses *resize* tidak memotong wajah.

Before Resize	After Resize
	

- *FlipItem*



Dengan flip ini, akan dilakukan pembalikan gambar terhadap sumbu x dan y secara bervariasi. Hal ini memungkinkan model mendapatkan lebih banyak variasi jika suatu bentuk wajah memiliki ciri pada salah satu sisi saja.

Before Flip	After Flip
	

- *Normalize.from_stats(*imagenet_stats)*

Sebelum data *image* dimasukkan ke dalam model *neural network*, data tersebut akan di normalisasi terlebih dahulu. Pada pemodelan kali ini, kami memutuskan untuk menggunakan statistik nilai normalisasi berdasarkan data ImageNet¹ yang digunakan oleh model kami. Data ImageNet merepresentasikan *image* manusia dalam kehidupan sehari-hari sehingga dipilih sebagai nilai normalisasi.

Dengan proses normalisasi, model dapat mempelajari nilai piksel dalam citra dengan lebih baik.

Before Normalize	After Normalize
	

¹ImageNet adalah koleksi data *image* untuk pembelajaran model AI yang dikembangkan oleh Stanford Vision Lab, Stanford University.

Seluruh proses augmentasi diimplementasikan dengan menggunakan *library* **FastAI** dan dilakukan agar *model* dapat melakukan *generalisasi* terhadap kemungkinan yang akan di prediksi nantinya.

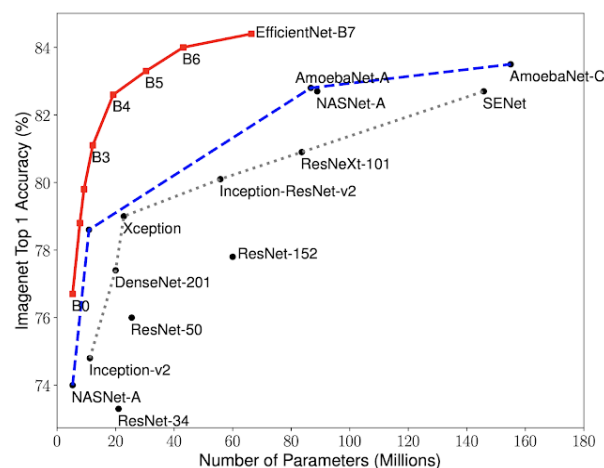
C. Training

1. Environment

Implementasi *training* dilakukan dengan menggunakan bahasa pemrograman Python versi 3.7.10 pada *cloud environment* Kaggle. Sumber daya komputasi adalah 4 core Intel Xeon CPU, 16GB RAM, NVIDIA Tesla P100, Debian GNU/Linux.

2. Arsitektur

Kami menggunakan EfficientNet-B4 dengan arsitektur EfficientNet yang memiliki akurasi yang baik dengan parameter yang lebih sedikit dibanding beberapa arsitektur model lainnya. Pada EfficientNet *scaling* dari *base model* EfficientNet-B0 dilakukan secara *compound scaling*. Selain itu, pada EfficientNet arsitektur *model* menggunakan *activation function* yang berbeda dari biasanya yaitu *swish function*.



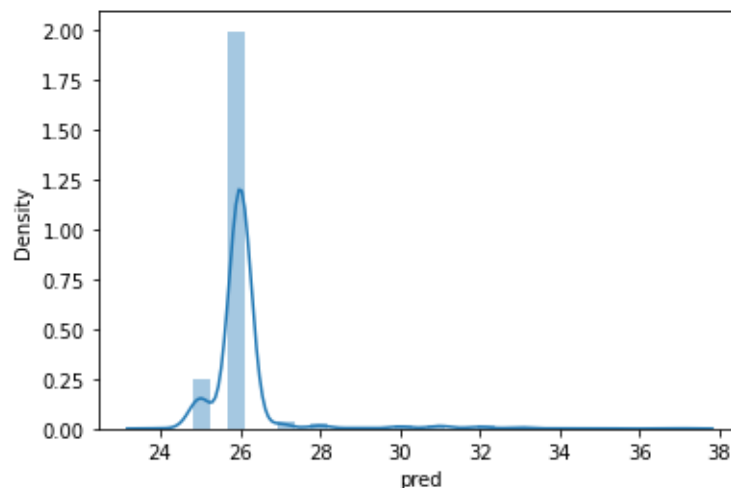
3. Hyperparameter

Kami menggunakan *optimization function* yaitu Adam (*Adaptive Moment Optimization*), metrik evaluasi MSE dan MAE pada fase *training*, dan *loss function* Huber's Loss. Dengan menggunakan Huber's Loss yaitu L1SmoothLoss yang memberikan hasil lebih baik daripada L1Loss dan L2Loss. Selain itu, kami juga menggunakan *optimization function* yaitu Adam Optimizer yang sudah dikenal memberikan performa lebih baik daripada Gradient Descent dan Stochastic Gradient Descent. Adam optimizer mencari optima lebih cepat daripada Gradient Descent dan Stochastic Gradient Descent.

D. Hasil

Hasil yang kami dapatkan dari proses *training* adalah nilai MSE terbaik yang didapatkan adalah 30. Berdasarkan eksplorasi yang telah kami lakukan hal ini dapat terjadi dikarenakan adanya persebaran umur yang kurang merata pada data *training*.

Setelah dilakukan prediction pada model test, kelompok kami mendapatkan hasil persebaran data test seperti gambar berikut. Terlihat hasil prediksi ada pada median dari data training dimana data training banyak pada range 25 - 27.



Daftar Pustaka

- [1] Howard, Jeremy, and Sylvain Gugger. *Deep Learning for Coders with Fastai and PyTorch*. "O'Reilly Media, Inc.," 2020.
- [2] Leblanc, Francois. "Leblancfg.com." *Leblancfg.com*, <https://leblancfg.com/autocrop/>. Accessed 1 Nov. 2021.
- [3] lukemelas. "Github - Lukemelas/EfficientNet-PyTorch: A PyTorch Implementation of EfficientNet and EfficientNetV2 (Coming Soon!)." *GitHub*, <https://github.com/lukemelas/EfficientNet-PyTorch>. Accessed 1 Nov. 2021.