オプティマイザ（最適化アルゴリズム）の利用方法

オプティマイザ（最適化アルゴリズム）はモデルをコンパイルする際に必要となるパラメータの1つです:

```
from keras import optimizers

model = Sequential()
model.add(Dense(64, kernel_initializer='uniform', input_shape=(10,)))
model.add(Activation('tanh'))
model.add(Activation('softmax'))

sgd = optimizers.SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mean_squared_error', optimizer=sgd)
```

上記の例のように，オプティマイザのインスタンスを `model.compile()` に渡す，もしくは，オプティマイザの名前を渡すことができます．後者の場合，オプティマイザのデフォルトパラメータが利用されます．

```
# オプティマイザを名前で指定すると，デフォルトパラメータが利用されます
model.compile(loss='mean_squared_error', optimizer='sgd')
```

Kerasのオプティマイザの共通パラメータ

`clipnorm` と `clipvalue` はすべての最適化法についてgradient clippingを制御するために使われます:

```
from keras import optimizers

# All parameter gradients will be clipped to
# a maximum norm of 1.
sgd = optimizers.SGD(lr=0.01, clipnorm=1.)
```

```
from keras import optimizers

# All parameter gradients will be clipped to
# a maximum value of 0.5 and
# a minimum value of -0.5.
sgd = optimizers.SGD(lr=0.01, clipvalue=0.5)
```

SGD

[source]

```
keras.optimizers.SGD(lr=0.01, momentum=0.0, decay=0.0, nesterov=False)
```

確率的勾配降下法オプティマイザ．

モーメンタム，学習率減衰，Nesterov momentumをサポートした確率的勾配降下法．

引数

- **lr**: 0以上の浮動小数点数．学習率．
- **momentum**: 0以上の浮動小数点数．モーメンタム．
- **decay**: 0以上の浮動小数点数．各更新の学習率減衰．
- **nesterov**: 真理値. Nesterov momentumを適用するかどうか．

RMSprop [source]

```
keras.optimizers.RMSprop(lr=0.001, rho=0.9, epsilon=None, decay=0.0)
```

RMSPropオプティマイザ．

デフォルトパラメータのまま利用することを推奨します． （ただし，学習率を除き，自由に調整可能です）

RMSPropはリカレントニューラルネットワークに対して良い選択となるでしょう．

引数

- **lr**: 0以上の浮動小数点数．学習率．
- **rho**: 0以上の浮動小数点数．
- **epsilon**: 0以上の浮動小数点数．微小量． `None` ならばデフォルトで `K.epsilon()` ．
- **decay**: 0以上の浮動小数点数．各更新の学習率減衰．

参考文献

- rmsprop: Divide the gradient by a running average of its recent magnitude

---

Adagrad [source]

```
keras.optimizers.Adagrad(lr=0.01, epsilon=None, decay=0.0)
```

Adagradオプティマイザ．

デフォルトパラメータのまま利用することを推奨します．

引数

- **lr**: 0以上の浮動小数点数．学習率．
- **epsilon**: 0以上の浮動小数点数． `None` ならばデフォルトで `K.epsilon()` ．
- **decay**: 0以上の浮動小数点数．各更新の学習率減衰．

参考文献

- Adaptive Subgradient Methods for Online Learning and Stochastic Optimization

---

Adadelta [source]

```
keras.optimizers.Adadelta(lr=1.0, rho=0.95, epsilon=None, decay=0.0)
```

Adadeltaオプティマイザ.

デフォルトパラメータのまま利用することを推奨します.

引数

- **lr**: 0以上の浮動小数点数. 学習率. デフォルト値を推奨します.
- **rho**: 0以上の浮動小数点数.
- **epsilon**: 0以上の浮動小数点数. 微小量. `None`ならばデフォルトで`K.epsilon()`.
- **decay**: 0以上の浮動小数点数. 各更新の学習率減衰.

参考文献

- Adadelta - an adaptive learning rate method

Adam [source]

```
keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=F
```

Adamオプティマイザ.

デフォルトパラメータは提案論文に従います.

引数

- **lr**: 0以上の浮動小数点数. 学習率.
- **beta_1**: 浮動小数点数, 0 < beta < 1. 一般的に1に近い値です.
- **beta_2**: 浮動小数点数, 0 < beta < 1. 一般的に1に近い値です.
- **epsilon**: 0以上の浮動小数点数. 微小量. `None`ならばデフォルトで`K.epsilon()`.
- **decay**: 0以上の浮動小数点数. 各更新の学習率減衰.
- **amsgrad**: 論文"On the Convergence of Adam and Beyond"にあるAdamの変種であるAMSGradを適用するかどうか.

参考文献

- Adam - A Method for Stochastic Optimization
- On the Convergence of Adam and Beyond

Adamax [source]

```
keras.optimizers.Adamax(lr=0.002, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0)
```

Adamaxは，Adamの提案論文の7節で提案されたAdamaxオプティマイザ．

これは無限ノルムに基づくAdamの拡張です．デフォルトパラメータは提案論文に従います．

引数

- **lr**: 0以上の浮動小数点数．学習率．
- **beta_1/beta_2**: 浮動小数点数, 0 < beta < 1. 一般的に1に近い値です．
- **epsilon**: 0以上の浮動小数点数．微小量． `None` ならばデフォルトで `K.epsilon()` ．
- **decay**: 0以上の浮動小数点数．各更新の学習率減衰．

参考文献

- Adam - A Method for Stochastic Optimization

Nadam [source]

```
keras.optimizers.Nadam(lr=0.002, beta_1=0.9, beta_2=0.999, epsilon=None, schedule_decay=0.00
```

Nesterov Adamオプティマイザ．

よく似たAdamはRMSPropとmomumtumを組み合わせたもので，NadamはRMSPropとNesterov momentumを組み合わせたものです．

デフォルトパラメータは提案論文に従います． デフォルトパラメータのまま利用することを推奨します．

引数

- **lr**: 0以上の浮動小数点数．学習率．
- **beta_1/beta_2**: 浮動小数点数, 0 < beta < 1. 一般的に1に近い値です．
- **epsilon**: 0以上の浮動小数点数．微小量． `None` ならばデフォルトで `K.epsilon()` ．

参考文献

- Nadam report
- On the importance of initialization and momentum in deep learning

---

TFOptimizer

[source]

```
keras.optimizers.TFOptimizer(optimizer)
```

TensorFlowのオプティマイザのためのラッパークラス.