model = Sequential([

  layers.Rescaling(1./255, input\_shape=(img\_height, img\_width, 1)),

  layers.Flatten(),

  layers.Dense(16, activation='tanh', name = "Dense1"),

  layers.Dense(num\_classes, activation = "softmax", name = "Dense2")

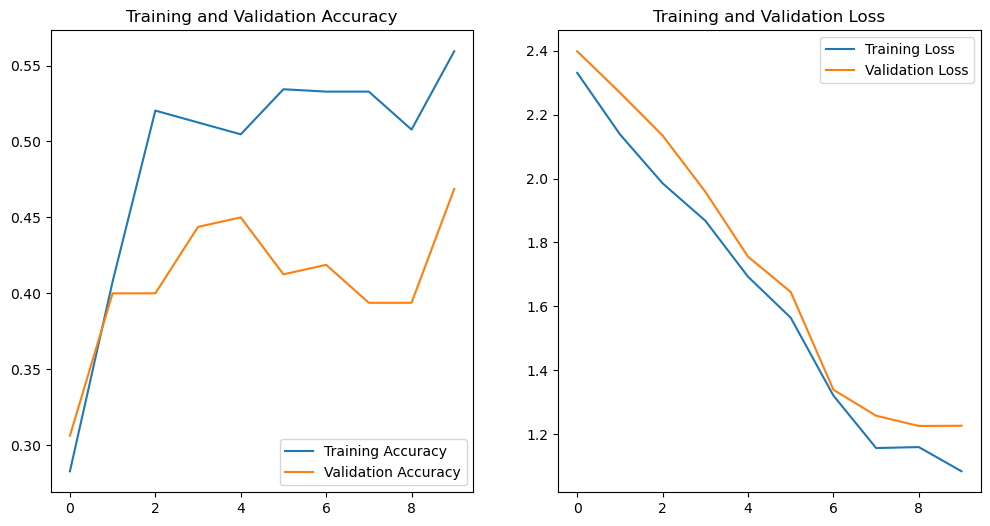
])

#Compiler le modèle

model.compile(optimizer='adam',

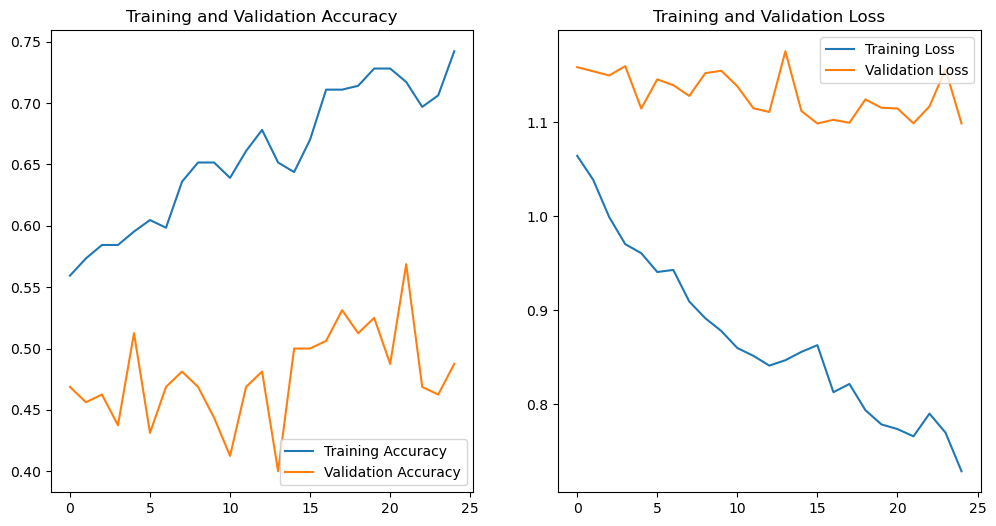
              loss="sparse\_categorical\_crossentropy",

              metrics=['accuracy'])



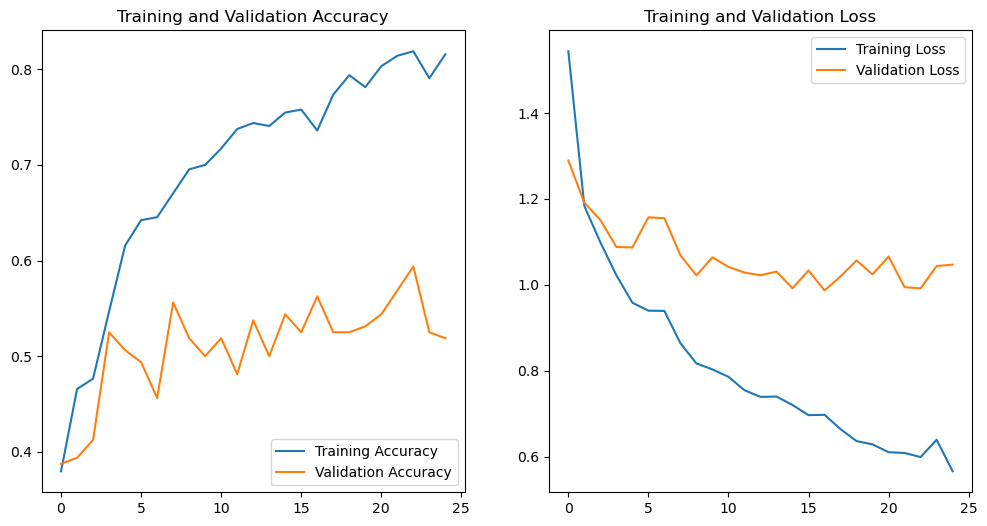
Avec 25 epochs :

* Val accuracy reste autour de 50%
* Training grimpe à 75% : overfitting



En ajoutant un DROPOUT :

* Training accuracy grimpe à 80%
* Validation accuracy stagne à 55%



Réduction des EPOCHS à 15

Agrandissement de la couche DENSE1 à 128 neurones => **OVERFITTING mais pas d’amélioration de l’accuracy sur la validation**

model = Sequential([

  layers.Rescaling(1./255, input\_shape=(img\_height, img\_width, 1)),

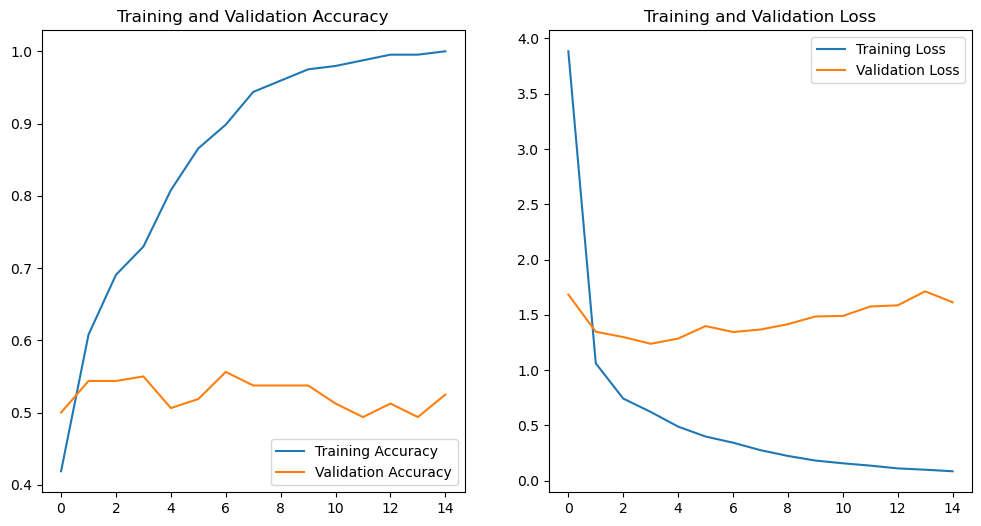
#  layers.Dropout(0.2),

  layers.Flatten(),

  layers.Dense(128, activation='relu', name = "Dense1"),

  layers.Dense(num\_classes, activation = "softmax", name = "Dense3")

])



**Ajout de neurones & Early stopping : 42% VAL ACCURACY**

model = Sequential([

  layers.Rescaling(1./255, input\_shape=(img\_height, img\_width, 1)),

#  layers.Dropout(0.2),

  layers.Flatten(),

  layers.Dense(128, activation='relu', name = "Dense2"),

  layers.Dense(64, activation='relu', name = "Dense3"),

  layers.Dense(8, activation='relu', name = "Dense5"),

  layers.Dense(num\_classes, activation = "softmax", name = "Dense6")

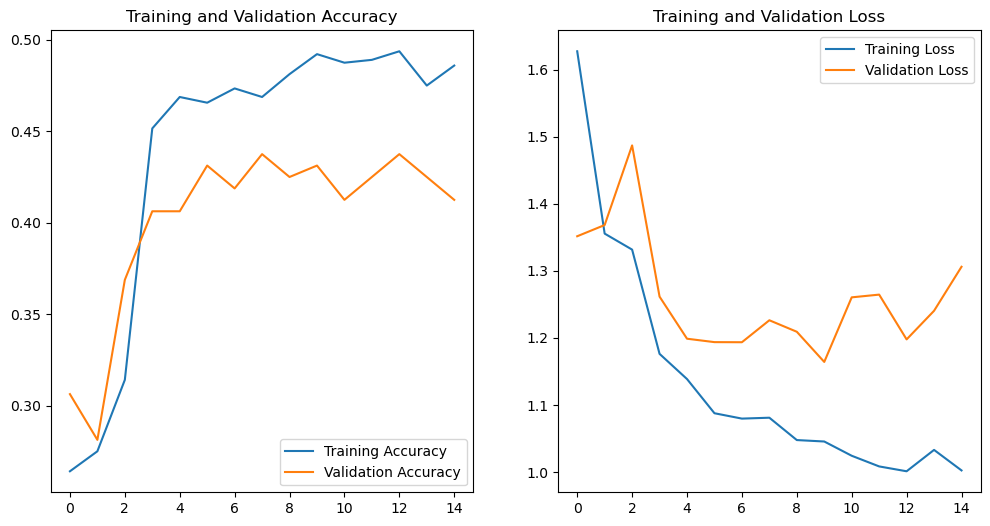
])

#Compiler le modèle

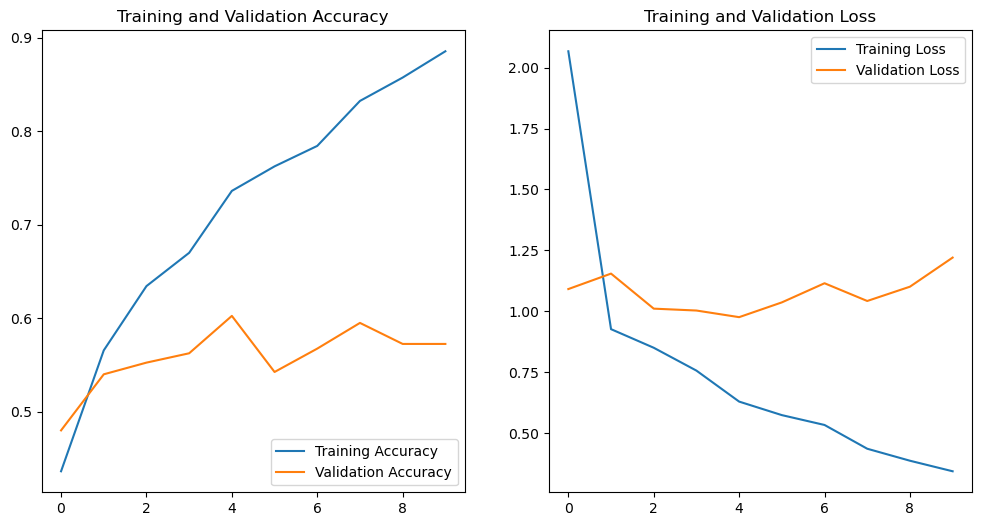
model.compile(optimizer='adam',

              loss="sparse\_categorical\_crossentropy",

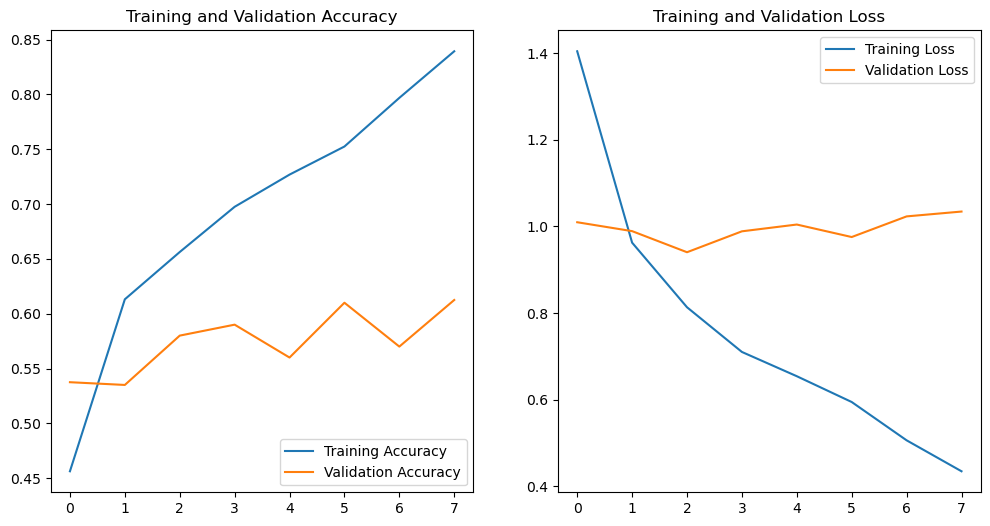
              metrics=['accuracy'])

****

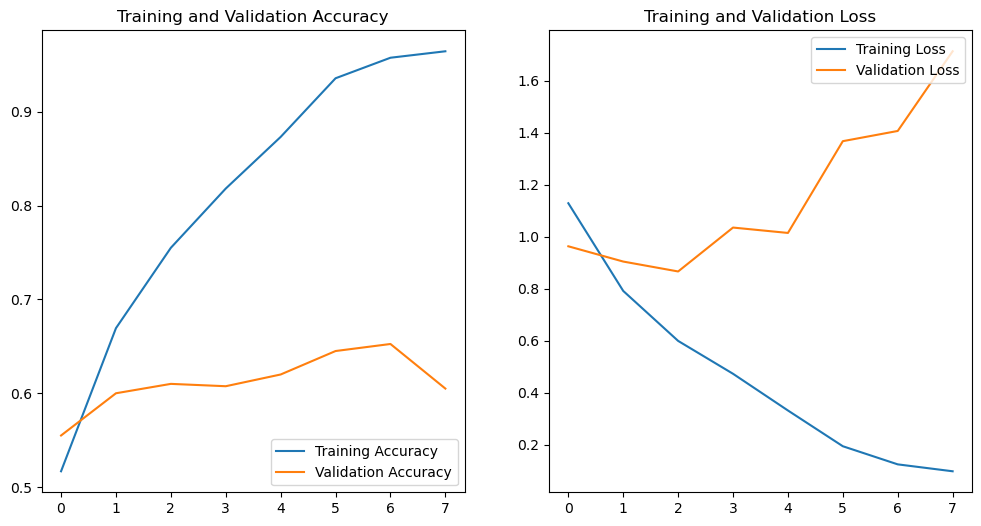
* **Test en augmentant la taille du dataset à 500 images par classe avec mêmes paramètres = 2000 images au lieu de 800 => Val\_accuracy = 45%**
* **Test en réduisant à 3 couches de neurones denses 128, 32, 4 => on monte à 55% VAL ACCURACY mais avec de l’overfitting**



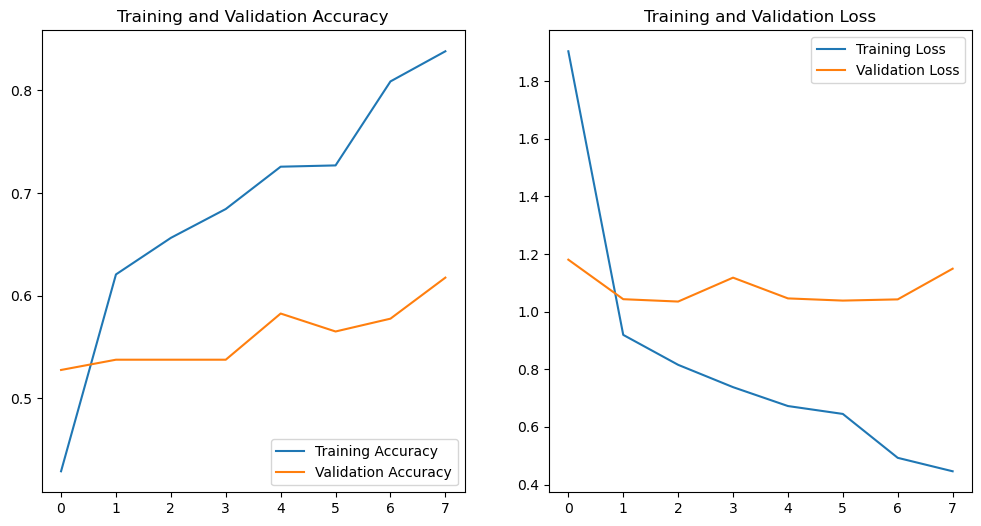
* **Avec un dropout, 60% VAL ACCURACY mais overfitting**

****

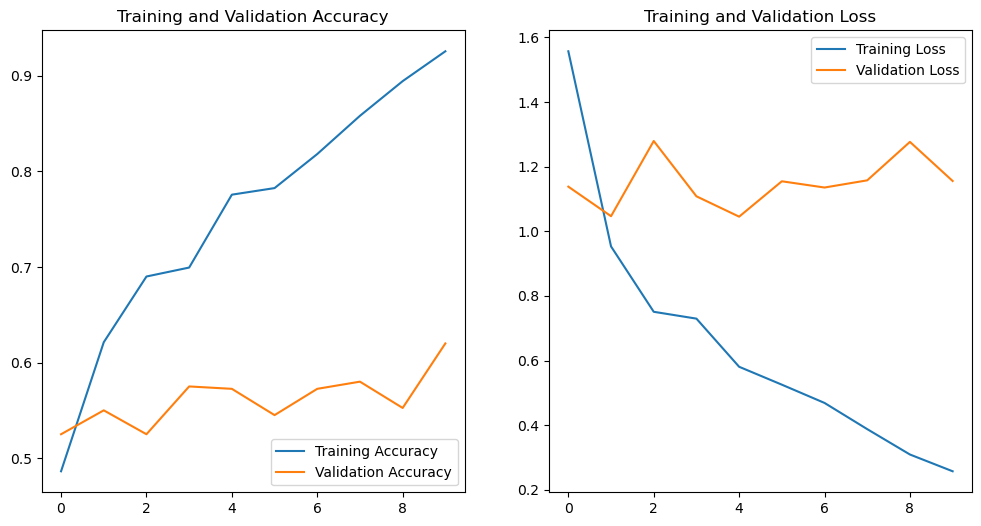
* **Avec 2 couches de convolution… on ne dépasse pas 60% par contre on gagne beaucoup en accuracy sur le jeu d’entraînement**

****

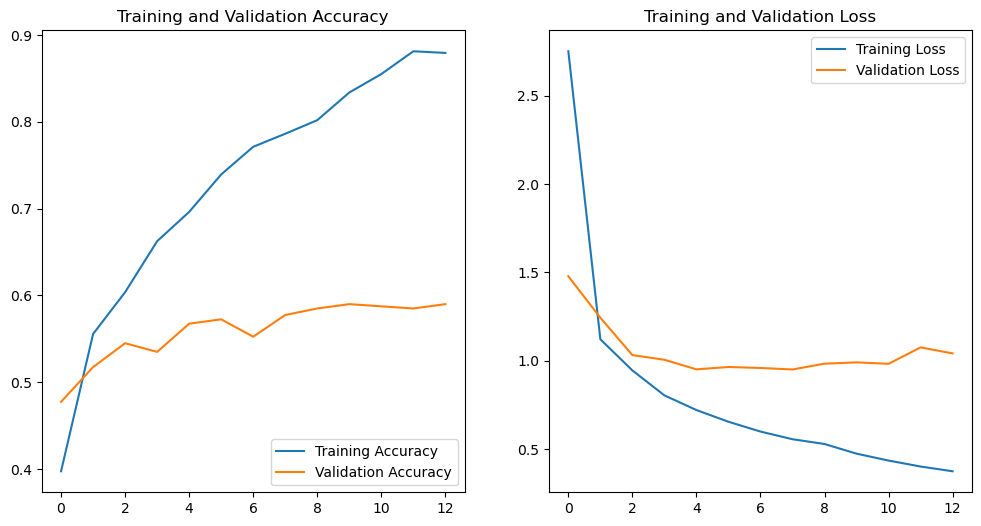
1. **Test en faisant varier le batch size**
   1. **2 couches denses cachées, batch size 32**
2. model = Sequential([
3. layers.Rescaling(1./255, input\_shape=(img\_height, img\_width, 1)),
4. #  layers.Dropout(0.2),
5. layers.Flatten(),
6. layers.Dense(128, activation='relu', name = "Dense2"),
7. layers.Dense(32, activation='relu', name = "Dense5"),
8. layers.Dense(num\_classes, activation = "softmax", name = "Dense6")
9. ])

****

* 1. **Même 2 couches denses cachées, batch size 64 (x2) => légère**

****

1. **Même 2 couches denses cachées, batch size 128 (x4) => même précision mais la val loss est plus haute**

****