J = theta \* x

dtheta = x

theta\_plus = theta + epsilon

theta\_minus = theta - epsilon

cost\_plus = forward\_propagation(x, theta\_plus)

cost\_mines = forward\_propagation(x, theta\_minus)

gradapprox = (cost\_plus - cost\_mines)/(2\*epsilon)

grad = backward\_propagation(x, theta)

numerator = np.linalg.norm(grad - gradapprox)

denominator = np.linalg.norm(grad) + np.linalg.norm(gradapprox)

difference = numerator/denominator

theta\_plus = np.copy(parameters\_values)

theta\_plus[i] = theta\_plus[i] + epsilon

J\_plus[i],\_ = forward\_propagation\_n(X, Y, vector\_to\_dictionary(theta\_plus))

theta\_minus = np.copy(parameters\_values)

theta\_minus[i] = theta\_minus[i] - epsilon

J\_minus[i],\_ = forward\_propagation\_n(X, Y, vector\_to\_dictionary(theta\_minus))

gradapprox[i] = (J\_plus[i] - J\_minus[i])/(2\*epsilon)

difference = np.linalg.norm(grad - gradapprox)/(np.linalg.norm(grad) + np.linalg.norm(gradapprox))