No Regularization

Dropout, Reg\_Kernel = 0

val\_loss > 0.005 (no convergence)

Group: optim

RMSprop 8

nadam 7

adam 6

\*adam slightly better at reaching convergence?

Group: units1

[200]: 6

[500]: 6

[100]: 5

[300]: 4

\*using 100 or 300 slightly better at reaching convergence?

\*using 500 seems not to improve

Group: dense\_mid\_size

[50]: 8

[100]: 5

[30]: 5

[10]: 3

\*avoid using [50]

\*it seems to be better use less?

###################################################################################

val\_loss < 0.005 and val\_loss > 0.001 (intermediate convergence)

Group: optim

RMSprop 4

nadam 4

adam 4

\*no conclusion

Group: units

[300]: 4

[500]: 3

[200]: 3

[100]: 2

\*no conclusion?

Group: dense\_mid\_size

[10]: 6

[30]: 4

[100]: 2

[50]: 0

\*no conclusion?

###################################################################################

val\_loss < 0.001 (better convergence)

Group: optim

RMSprop 4

nadam 5

adam 6

\*it seems to be better using adam

Group: units

[300]: 4

[500]: 3

[200]: 3

[100]: 5

\*no conclusion?

Group: dense\_mid\_size

[10]: 3

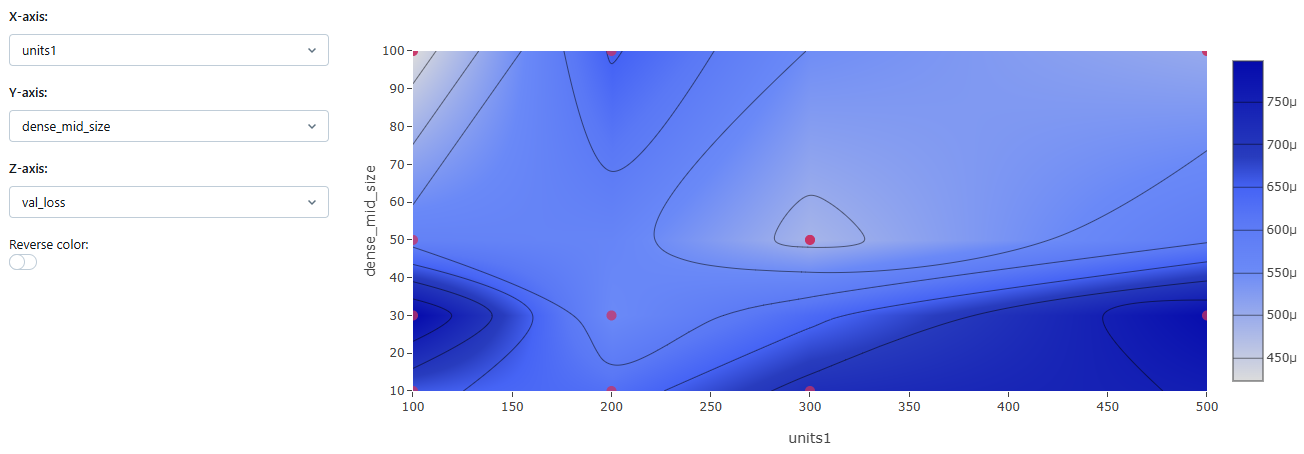
[30]: 3

[100]: 5

[50]: 4

\*different to "no convergence" results

Contour Plot for 15 models with val\_loss < 0.001



Top 5 models

R2score = 0.77 (units = 100, dense\_mid\_size = 100, optim = nadam)

R2score = 0.736 (units = 300, dense\_mid\_size = 50, optim = RMSprop)

R2score = 0.726 (units = 500, dense\_mid\_size = 100, optim = nadam)

R2score = 0.699 (units = 200, dense\_mid\_size = 30, optim = nadam)

R2score = 0.685 (units = 100, dense\_mid\_size = 50, optim = adam)

Top 5 models with units = 100

R2score = 0.77 (units = 100, dense\_mid\_size = 100, optim = nadam)

R2score = 0.685 (units = 100, dense\_mid\_size = 50, optim = adam)

R2score = 0.642 (units = 100, dense\_mid\_size = 10, optim = nadam)

R2score = 0.593 (units = 100, dense\_mid\_size = 100, optim = adam)

R2score = 0.566 (units = 100, dense\_mid\_size = 30, optim = adam)

Conclusion

Optim = [adam, nadam] better for low complex models (less units1)

Proceed with less complex models, fix units1 = 100 and keep varying dense\_mid\_size

Testing Regularization Parameters with less complex structure

Top 10 models

R2score = 0.548 (dropout=0.15, kernel\_l2 = 1e-5, optim = nadam)

R2score = 0.540 (dropout=0.15, kernel\_l2 = 1e-5, optim = adam)

R2score = 0.523 (dropout=0.20, kernel\_l2 = 1e-5, optim = nadam)

R2score = 0.550 (dropout=0.02, kernel\_l2 = 1e-4, optim = adam)

R2score = 0.514 (dropout=0.15, kernel\_l2 = 1e-4, optim = adam)

R2score = 0.493 (dropout=0.05, kernel\_l2 = 1e-4, optim = nadam)

Sequence Length and Batch Size + Regularization Fine Tuning

R2score = 0.762 (seq\_length=10, batch\_size = 10, kernel\_l2 = 1e-3, optim = nadam)

R2score = 0.737 (seq\_length=10, batch\_size = 30, kernel\_l2 = 1e-5, optim = nadam)

R2score = 0.539 (seq\_length=30, batch\_size = 30, kernel\_l2 = 1e-3, optim = adam)

R2score = 0.496 (seq\_length=30, batch\_size = 10, kernel\_l2 = 1e-3, optim = nadam)

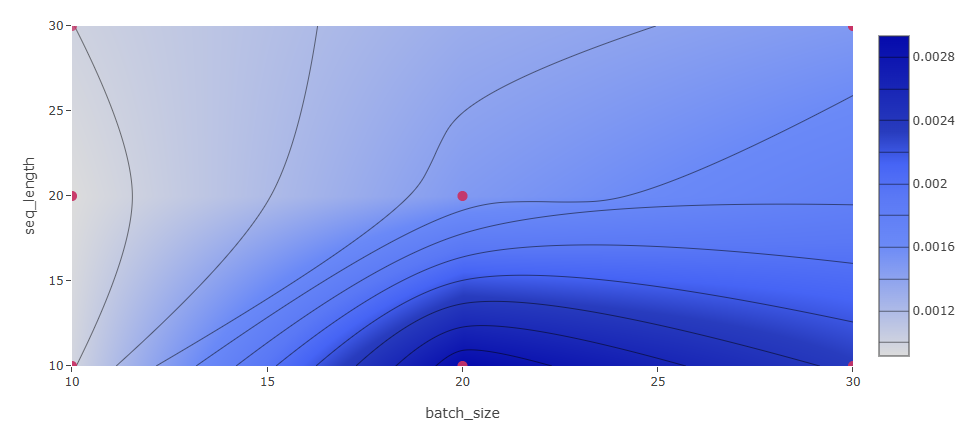
R2score = 0.494 (seq\_length=30, batch\_size = 30, kernel\_l2 = 1e-5, optim = adam)

R2score = 0.465 (seq\_length=30, batch\_size = 10, kernel\_l2 = 1e-5, optim = adam)

Conclusion: use smaller seq\_length and batch\_size, use kernel\_l2 = 0.001

optim = nadam tends to be better

Sequence Length and Batch Size



Conclusion: Use smaller batch\_size

R2score = 0.802 (seq\_length=20, batch\_size = 10, optim = nadam)

R2score = 0.766 (seq\_length=20, batch\_size = 10, optim = adam)

R2score = 0.762 (seq\_length=10, batch\_size = 10, optim = nadam)

R2score = 0.737 (seq\_length=10, batch\_size = 30, optim = nadam)

R2score = 0.611 (seq\_length=20, batch\_size = 20, optim = adam)

Seq length of 20 tends to be better

Fixing batch\_size = 10

R2score = 0.754 (seq\_length=5, optim = nadam)

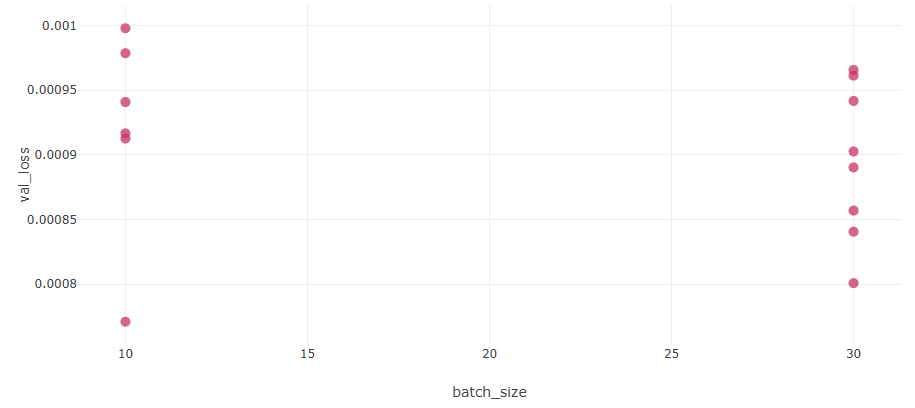
R2score = 0.610 (seq\_length=5, optim = adam)

R2score = 0.501 (seq\_length=15, optim = adam)

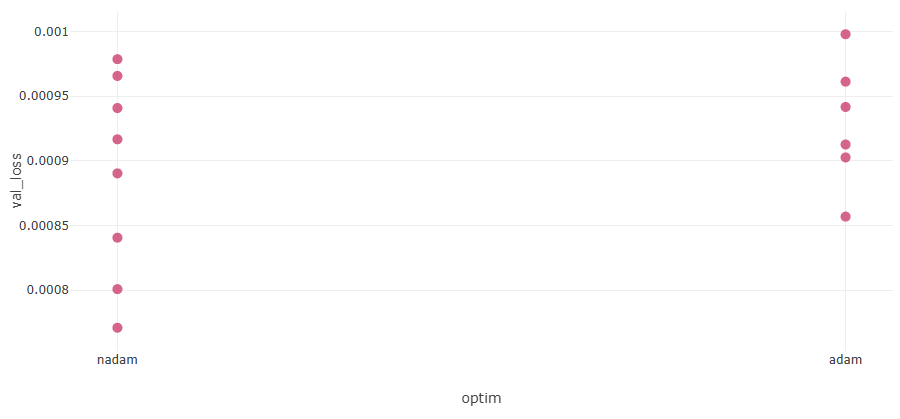
R2score = 0.437 (seq\_length=15, optim = nadam)

Conclusions Mixing Data from Different Experiments

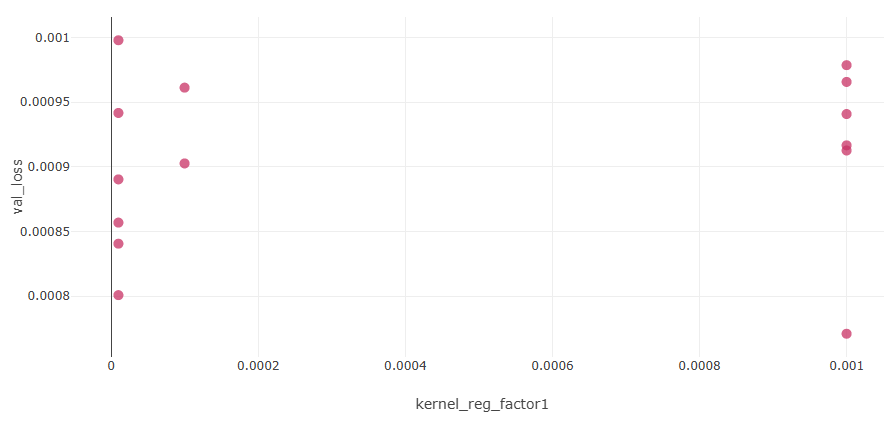
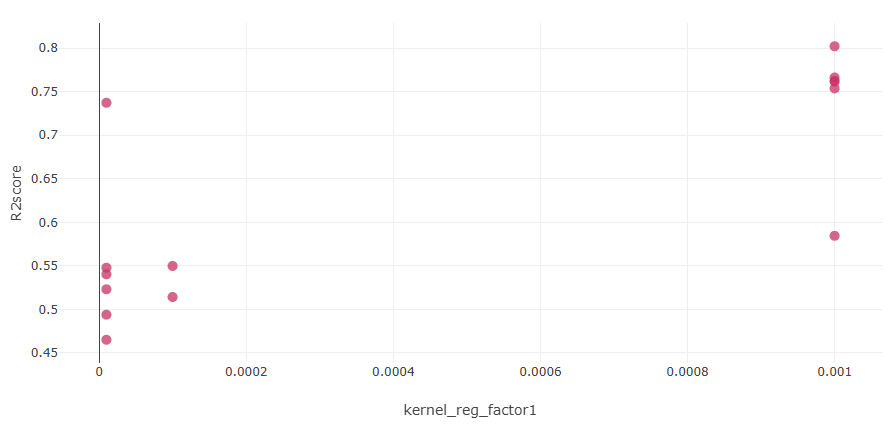
Batch\_size: fix in 10 improves chance to find champion model

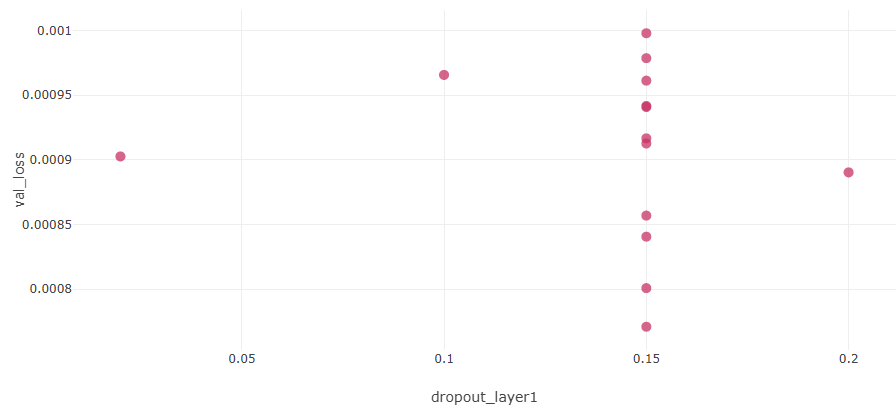
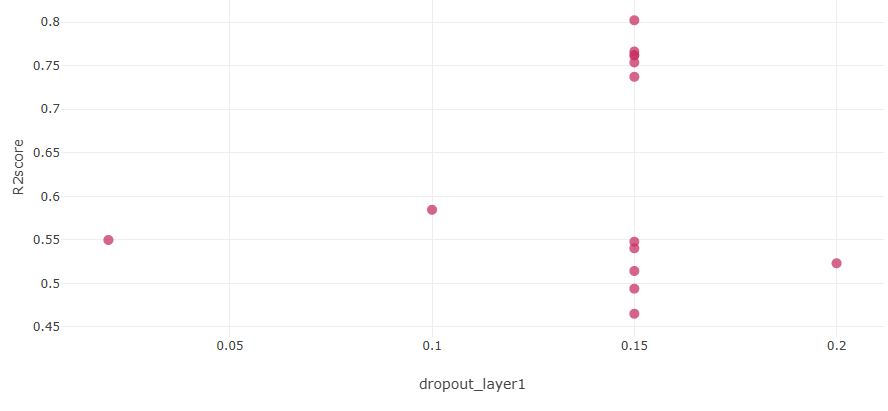
Optim: nadam improves chance to find the champion model

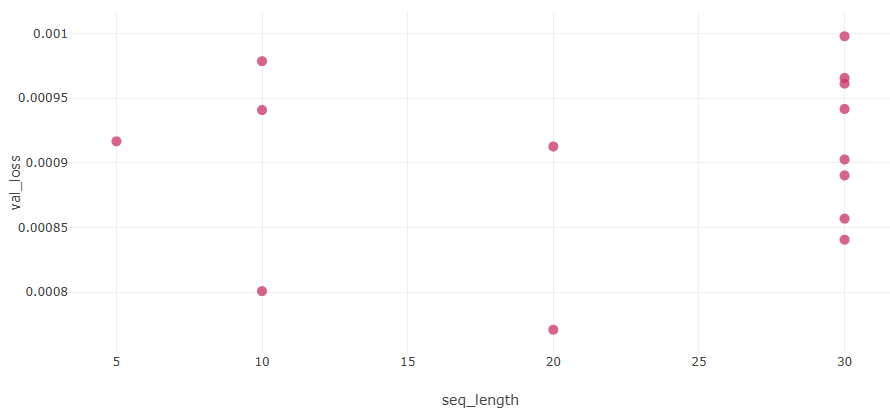
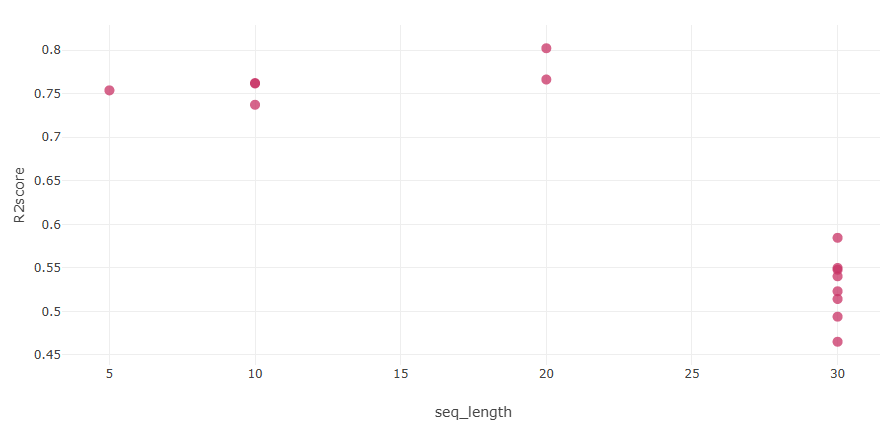
Kernel\_l2: use 0.001

Dropout: use 0.15

Sqe\_length: optimal value seems to be between 10 and 30, possibly 15, 20 or 25

Test 8: Revamp “Sigmoid” as activation function

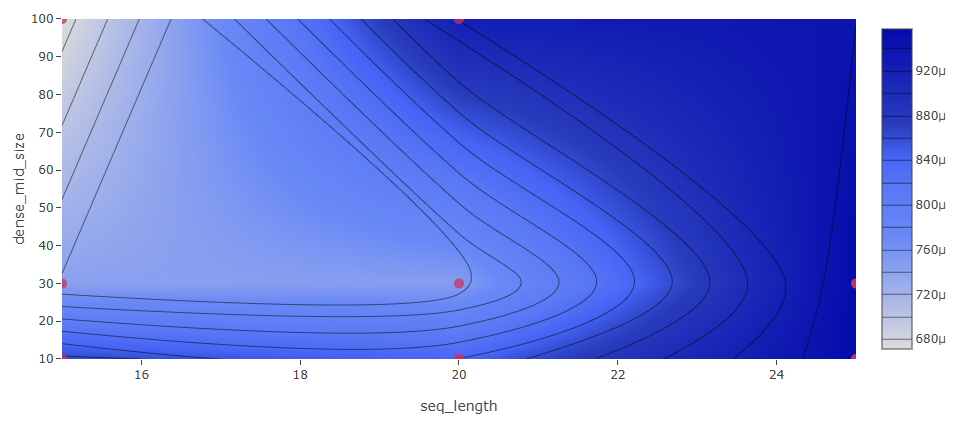
Val\_loss in [0.0035, 0.94]

max(R2score) = 0,35

Conclusion: Sigmoid not worthy

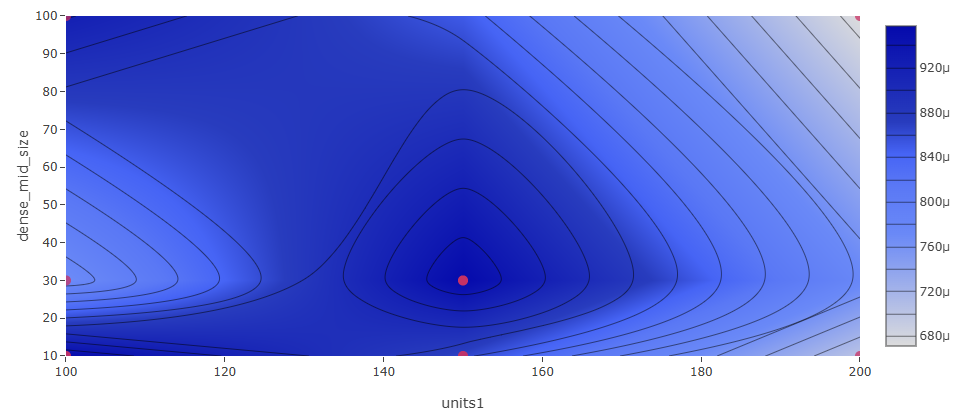
Test 8: limiting to relu activation

Seq\_length x dense\_mid\_size



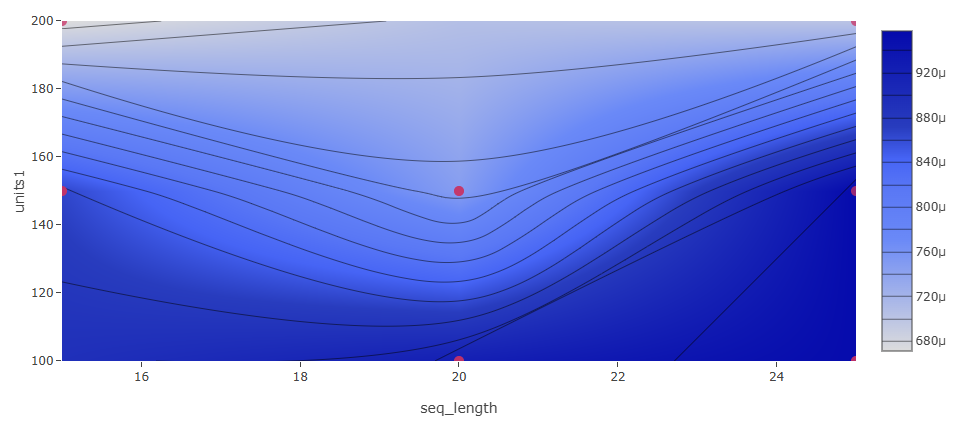
Conclusion: better to combine “less seq\_length” (around 15) and more “dense\_mid\_size” (from 30 to 100)

Units1 x dense\_mid\_size



Conclusion: in the extrems of dense\_mid\_size (10 or 100) combine with more units1 (200). When dense\_mid\_size is in the middle (30) combine with less units1.

Seq\_length x Units1



Conclusion: combine middle seq\_length with middle units1 (20 and 150), and extrems (15, 25) combine with more units (200)

Top 8 Models

R2score = 0.832, val\_loss = 6,71e-4 (units=200, dense\_mid = 100, seq\_length=15)

R2score = 0.812, val\_loss = 7,427e-4 (units=150, dense\_mid = 30, seq\_length=15)

R2score = 0.810, val\_loss = 7,115e-4 (units=150, dense\_mid = 10, seq\_length=20)

R2score = 0.806, val\_loss = 7,471e-4 (units=150, dense\_mid = 30, seq\_length=20)

R2score = 0.785, val\_loss = 7,011e-4 (units=200, dense\_mid = 10, seq\_length=25)

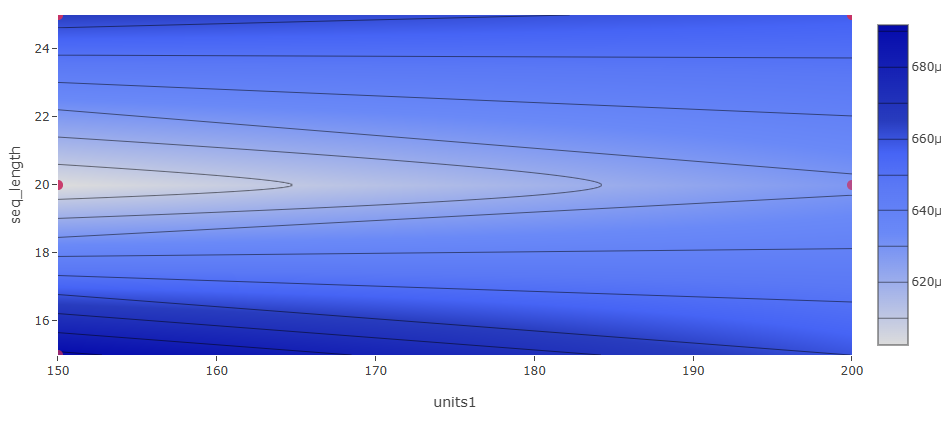
R2score = 0.778, val\_loss = 8,648e-4 (units=150, dense\_mid = 10, seq\_length=15)

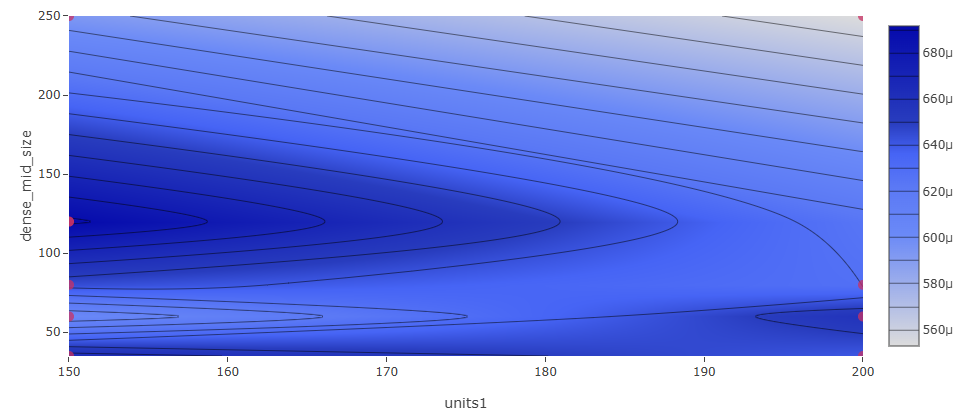
R2score = 0.765, val\_loss = 8,407e-4 (units=100, dense\_mid = 10, seq\_length=20)

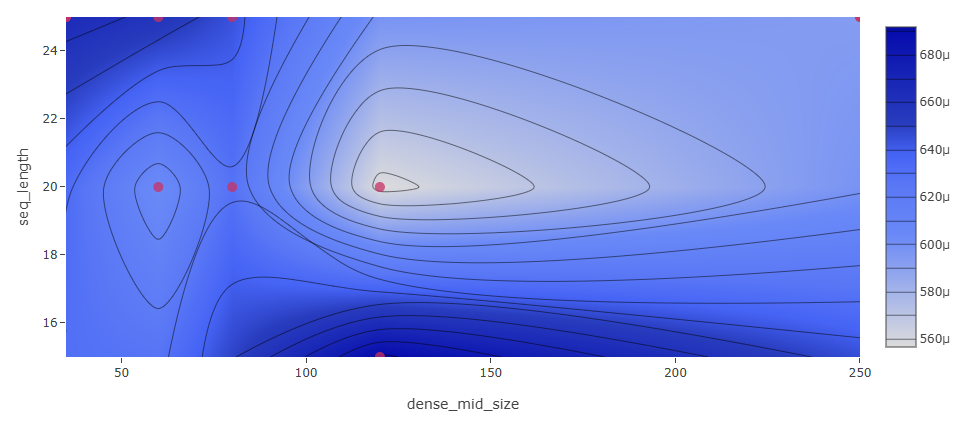
R2score = 0.750, val\_loss = 7,662e-4 (units=100, dense\_mid = 30, seq\_length=25)

Conclusion: remove units1=100

Test 9: Checking combinations of units1, dense\_mid and seq\_length







Top 8 Models

R2score = 0.857, val\_loss = 5,566e-4 (units=150, dense\_mid = 120, seq\_length=20)

R2score = 0.846, val\_loss = 6,025e-4 (units=150, dense\_mid = 60, seq\_length=20)

R2score = 0.833, val\_loss = 6,281e-4 (units=200, dense\_mid = 80, seq\_length=20)

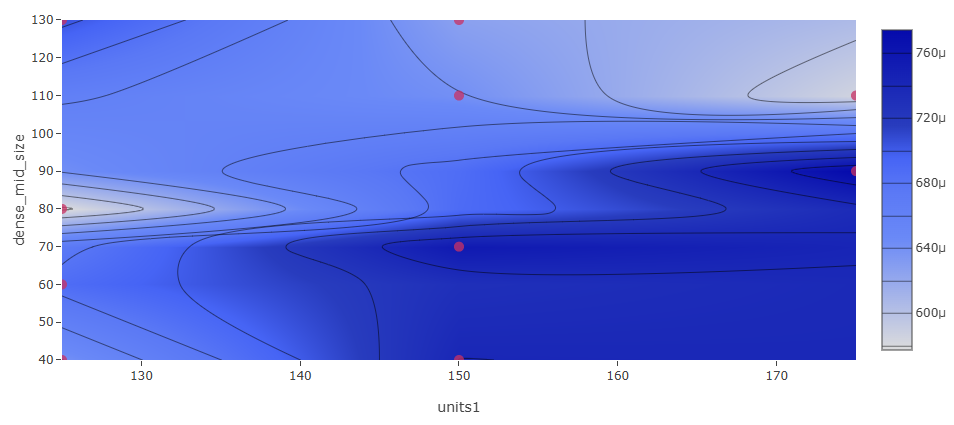
R2score = 0.831, val\_loss = 6,918e-4 (units=150, dense\_mid = 120, seq\_length=15)

R2score = 0.823, val\_loss = 5,529e-4 (units=200, dense\_mid = 250, seq\_length=25)

R2score = 0.819, val\_loss = 5,931e-4 (units=150, dense\_mid = 250, seq\_length=25)

Better using units1 around 150, dense\_mid around 60, 80, 120 and seq\_length around 20

Test 10



R2score = 0.856, val\_loss = 5,773e-4 (units=125, dense\_mid = 80, seq\_length=20)

Test 11

Champion Model

R2score = 0.857, val\_loss = 5,678e-4 (units=130, dense\_mid = 80, seq\_length=20)

Old analysis.txt

R2-Score positive

R2-Score > 0.5

validation\_loss < 0.2

RMSE < 2

loss (training) < 0.1 for convergence

RMSE > 2

activation1 : 11 relu vs. 27 sigm

optim : 17 adam vs. 21 SGD

dropout1 : 16 [0.1] vs. 22 None

reg\_kernel1 : 22 [0.01] vs. 16 None

Dense\_mid: 22 [10] vs. 16 None

RMSE > 5

\*Check activation function

--------------------------------

sigmoid + adam = 11

sigmoid + SGD = 16

relu + SGD = 4

relu + adam = 1

\*better to use adam

\*better to use relu

--------------------------------

sigmoid + dropout[0.1] = 12

sigmoid + dropout[None] = 15

relu + dropout[0.1] = 1

relu + dropout[None] = 4

\*better to use dropout = 0.1

\*better to use relu

--------------------------------

sigmoid + reg\_kernel[0.01] = 16

sigmoid + reg\_kernel[None] = 11

relu + reg\_kernel[None] = 3

relu + reg\_kernel[0.01] = 2

\*better to use relu

\*unclear to use reg\_kernel or not

\*Check optim

--------------------------------

adam + droput[0.1] = 4

SGD + droput[0.1] = 9

adam + droput[None] = 8

SGD + droput[None] = 11

\*better to use adam

--------------------------------

adam + reg\_kernel[0.01] = 8

SGD + reg\_kernel[0.01] = 10

adam + reg\_kernel[None] = 4

SGD + reg\_kernel[None] = 10

\*better to use adam

\*reg\_kernel unclear

\*Check optim: use adam

--------------------------------

RMS > 5

adam = 12

SGD = 20

RMSE < 0.7

adam = 2

SGD = 0

\*Check activation: use relu

--------------------------------

RMS > 5

relu = 5

sigmoid = 27

RMSE < 0.7

relu = 2

sigmoid = 0

\*Check dropout: use dropout, check more values

--------------------------------

RMS > 5

[0.1] = 13

None = 19

RMSE < 0.7

[0.1] = 2

None = 0

\*Check dense\_mid\_size: unclear, check more values

--------------------------------

RMS > 5

[10] = 16

None = 16

RMSE < 0.7

[10] = 2

None = 0

\*Check n\_inputs: unclear, check again

--------------------------------

RMS > 5

[1] = 15

[3] = 17

RMSE < 0.7

[1] = 2

[3] = 0

\*Check reg\_kernel: unclear, check again

--------------------------------

RMS > 5

[0.01] = 18

None = 14

RMSE < 0.7

[0.01] = 1

None = 1

################################################

\*Check reg\_kernel: better low values or None

--------------------------------

RMS > 5

[0.1] = 4

[None, 0.01, 0.001] = 0

RMSE < 0.7

[0.1] = 0

[None, 0.01, 0.001] = 4

\*best kernel (l2) = 0.001

\*Check dropout: inconclusive for which value, tending to 0.1

--------------------------------

RMS > 5

[0.1] = 2

[0.05, 0.2] = [1, 1]

RMSE < 0.7

[0.05] = 3

[0.1] = 1

[0.2] = 2

\*Check dense\_mid\_size: better None or More instead of few, check more values NOT None

--------------------------------

RMS > 5

[None, 10] = 1

[5] = 2

[20] = 0

RMSE < 0.7

[None] = 2

[5, 10] = 1

[20] = 2

\*Check n\_inputs: unclear, check again

--------------------------------

RMS > 5

[1] = 2

[3] = 2

RMS > 2

[1] = 21

[3] = 16

RMS > 1

[1] = 38

[3] = 39

RMSE < 0.7

[1] = 5

[3] = 1

\* also similar in terms of valid loss

\* in terms of RMSE, best models tending to [1] input

################################################

\*Check n\_inputs: best 1 input only

--------------------------------

RMS > 5

0

RMSE < 0.7

[1] = 3

[3] = 0

\*Check dropout: better with 0.1

--------------------------------

RMS > 5

0

RMSE < 0.7

[0.05, 0.2] = 0

[0.1] = 2

[0.15] = 1

\*Check dense\_mid\_size: better None or More instead of few, check more values NOT None

--------------------------------

RMS > 5

0

RMSE < 0.7

[None] = 1

[20] = 0

[30] = 1

[40] = 1

################################################

\*Check optim:

--------------------------------

inputs = 3:

adam : 0.2 < R2 < 0.65

nadam : 0.7 < R2 < 0.85, 1 outlier -0.1

RMSprop : 0.6 < R2 < 0.85, 1 outlier -0.3 at (units = 200, dense\_mid = 50, inputs = 3)

adam : 0.0022 < valid\_loss < 0.0057

nadam : 0.0017 < valid\_loss < 0.0035

RMSprop : 0.0017 < valid\_loss < 0.0042

adam : 1.2 < RMSE < 1.8

nadam : 0.6 < RMSE < 1, 1 outlier 2.1 at (units = 200, dense\_mid = None, inputs = 3)

RMSprop : 0.6 < RMSE < 1.2, 1 outlier 2.1 at (units = 200, dense\_mid = 50, inputs = 3)

inputs = 1:

adam : -0.8 < R2 < -0.3

nadam : 0.1 < R2 < 0.9, 1 outlier -0.6

RMSprop : 0.1 < R2 < 0.6, 1 outlier -0.4 at (units = 200, dense\_mid = 50, inputs = 3)

adam : 0.0027 < valid\_loss < 0.0042

nadam : 0.0015 < valid\_loss < 0.0035

RMSprop : 0.0017 < valid\_loss < 0.0040, 1 outlier 0.0085

adam : 2.3 < RMSE < 2.7

nadam : 0.6 < RMSE < 2.6, 1 outlier 2.1 at (units = 200, dense\_mid = None, inputs = 3)

RMSprop : 1.4 < RMSE < 2.4, 1 outlier 2.1 at (units = 200, dense\_mid = 50, inputs = 3)

discard adam

\*best models according to RMSE and R2-Score

(units=, optim=, dense\_mid=, inputs=) -> [R2=, RMSE=]

(units=200, optim=nadam, dense\_mid=50, inputs=3) -> [R2=0.824, RMSE=0.82]

(units=100, optim=nadam, dense\_mid=50, inputs=3) -> [R2=0.885, RMSE=0.664]

(units=100, optim=RMSprop, dense\_mid=50, inputs=3) -> [R2=0.823, RMSE=0.824]

(units=100, optim=RMSprop, dense\_mid=None, inputs=3) -> [R2=0.886, RMSE=0.661]

(units=200, optim=nadam, dense\_mid=50, inputs=1) -> [R2=0.81, RMSE=0.874]

(units=100, optim=nadam, dense\_mid=50, inputs=1) -> [R2=0.899, RMSE=0.637]

################################################

\*Check batch\_size: discard batch\_size = 10

--------------------------------

RMSE > 5

[10] : 2

[20,30] : 1

RMSE < 0.7

[10] : 0

[20] : 3

[30] : 4

\*Check seq\_length: test more for [20,30]

--------------------------------

RMSE > 5

[10] : 2

[20,30] : 1

RMSE < 0.7

[10] : 0

[20] : 3

[30] : 4

################################################

CHAMPION MODELS

################################################

Exp 9

fun-bass-942

trusting-crane-275

Exp 8

unleashed-bird-189

Exp 7

carefree-snipe-956

Exp 6

calm-yak-301

#########################

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

NEW ANALYSIS

val\_loss > 0.005 (no convergence)

Group: optim

RMSprop 8

nadam 7

adam 6

\*adam slightly better at reaching convergence?

Group: units1

[200]: 6

[500]: 6

[100]: 5

[300]: 4

\*using 100 or 300 slightly better at reaching convergence?

\*using 500 seems not to improve

Group: dense\_mid\_size

[50]: 8

[100]: 5

[30]: 5

[10]: 3

\*avoid using [50]

\*it seems to be better use less?

###################################################################################

val\_loss < 0.005 and val\_loss > 0.001 (intermediate convergence)

Group: optim

RMSprop 4

nadam 4

adam 4

\*no conclusion

Group: units

[300]: 4

[500]: 3

[200]: 3

[100]: 2

\*no conclusion?

Group: dense\_mid\_size

[10]: 6

[30]: 4

[100]: 2

[50]: 0

\*no conclusion?