Einige SNNS-features zur Info – nicht unbedingt nötig

[PROJECTION:] ermöglicht, die Aktivierung eines Neurons in Abhängigkeit von zwei anderen Neuronen zu plotten, d.h. man kann die entstehenden Spiralen sehr schön sehen. Unter setup einstellen, die Aktivierung welcher Neuronen in welchem Bereich man als x/y-Achse (z.B. Eingaben) bzw. Farbdarstellung (z.B. Ausgabe) sehen will.

[CASCADE CORRELATION:] Minimalstartnetz (= Eingaben und Ausgaben) erzeugen,
unter control: Lernfunktion CC, Initialisierung CC_Weights, Update CC_Order. Cycles= maximale Anzahl hinzuzufügender Hidden-Units. Die Parameter der Lernfunktion werden in den Fenstern neben LEARN eingestellt. Die Funktion selber und alle übrigen Parameter werden im Fenster CASCADE (entsprechenden Knopf im manager drücken) gewählt. Unter CASCADE einzustellen:
maximaler Fehler, der je Ausgabeeinheit toleriert wird, so daß CC abbricht;
Lernfunktion für das Maximieren der Kovarianz (f.K.) und Tainieren des Perzeptron (f.P.):
Quickprop (Lernrate für Perzeptronlernen z.B. 0.0001, maximales Wachstum f.P. z.B. 2, Weight Decay f.P. z.B. 0.0001, Lernrate für Kovarianzlernen z.B. 0.0007, Wachstum f.C.), Backprop (Lernrate f.P., Momentum f.P., Flat-spot-Elimination, Lernrate f.K., Momentum f.K.), Rprop (Verkleinerungsfaktor der Gewichtsänderung b.P. z.B. 0.5, Vergrößerung z.B. 1.2, nicht benutzt, Verkleinerung f.K., Vergrößerung f.K.).
Zusätzliche Knöpfe ermöglichen, die Anzahl der Neuronen und der Verbindungen zu minimieren (besser nicht probieren, bugs!). Es werden immer mehrere Kandidaten trainiert und der beste eingefügt. Einzustellen: maximale Anzahl an Kandidaten; Aktivierungsfunktion der Kandidaten; den minimalen Anteil vom alten Wert, den die Kovarianz sich beim Training ändern muß, um nicht das Kovarianztraining abzubrechen; die Kandidate patience, d.h. die Schritte, nach denen alle Kandidaten die Kovarianz berechnen; die maximale Anzahl von Kovarianzberechnungen, bevor das Kovarianztraining abgebrochen wird.
Analoge Abbruchkriterien gibt es beim Fehler bzgl. des Perzeptronlernens, diese werden im dritten Teil des Fensters eingestellt: Minimaler Anteil einer Fehleränderung, bevor das Training abgebrochen wird, Anzahl Schritte, bevor der Fehler berechnet wird, maximale Anzahl Schritte.
Lernen: mit ALL im control-panel, vorher INIT, das löscht die Hidden-Neuronen und initialisiert die Gewichte, man kann mit ALL auch weiterlernen und dann neue Hidden-Neuronen einfügen.

[PRUNING] Im control-panel die Lernfunktion auf PruningFeedForward stellen. In den fünf (nicht alle benötigten) Fenstern werden der Reihe nach die Parameter für das gewählte Lernverfahren zum Nachtrainieren eingegeben. INIT, SHUFFLE wie üblich, ALL startet das Pruning. Alle weiteren Parameter werden von einem Extramenu aus, Knopf PRUNING eingegeben. Die Parameter werden nach Schließen des Fensters aktualisiert.
Die Parameter im einzelnen: (Zu Beginn auf Defaultwerte gesetzt, der Reihe nach, [..] referiert auf einzugebende Werte.)
Unter dem obersten Select wählt man das Verfahren aus (MagPruning: Die Verbindung mit kleinstem Gewicht wird geprunt, OBD, OBS: OBD mit kompletter Hessematrix, Skeletonization, Noncontributing_Units: Neuronen ohne Output-Änderung bzw. mit Output parallel zu einer anderen Unit des Layers werden geprunt). Es wird je ein Link geprunt und nachtrainiert, bis der Fehler um [..] Prozent oder der SSE absolut um [..] wächst. YES/NO bestimmt, ob der letzte Schritt rückgängig gemacht wird.
Bei SELECT wählt man die Trainingsfunktion aus, es wird zu Beginn [..] Schritte trainiert und nach jedem Pruning [..] Schritte nachtrainiert, sofern nicht schon eher der SSE kleiner als [..] ist.
OBS benötigt Startwerte für die Diagonalelemente der Hessematrix.
Bei Skeletonization kann man angeben, ob Hidden/Input Units geprunt werden sollen.
Achtung, bug, wenn alle Neuronen weggeprunt werden, meckert SNNS! Aber dann ist sowieso was an der zu lernenden Funktion faul ...