Source:

<https://www.researchgate.net/figure/Feature-extraction-from-SLIM-images-a-extracting-the-texton-dictionary-from-a-training_fig5_315716935>

<https://www.researchgate.net/publication/308393344_Automated_brain_tumour_detection_and_segmentation_using_superpixel-based_extremely_randomized_trees_in_FLAIR_MRI>

<https://www.researchgate.net/publication/358702285_Deep_Learning_Approaches_to_Image_Texture_Analysis_in_Material_Processing>

**Model to classify in different nuclei classes**

Unsupervised because we do not know how the nuclei classes look like (smooth, strati ect.). To solve this task, we must assume that the Features mentioned below of a TF is more or less constant. We choose some nuclei and aggregate them together to one big picture (See question below). We have to be careful not to take any outliers.

**Feature Extraction for each nucleus:**

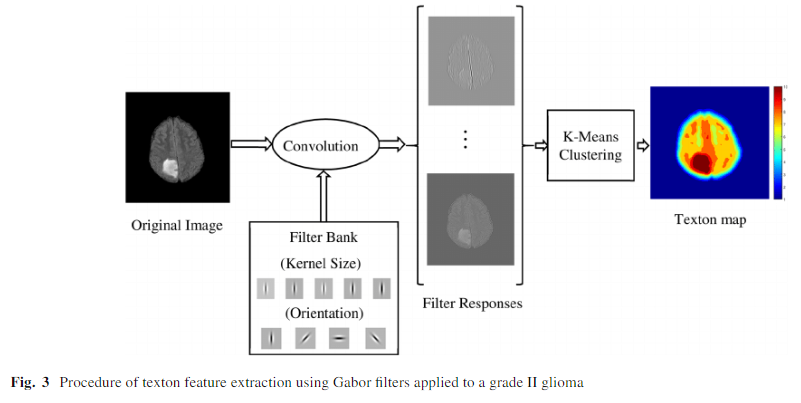
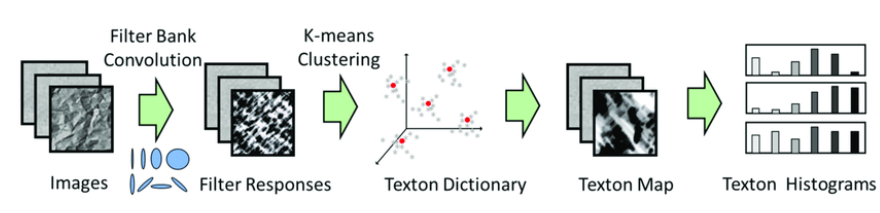
Based on intensity: (Remember also that comparing whole nucleus intensities of Ypet between different transcription factors is meaningless, since these variations stem from technical issues -> first normalize)

* Moments: average, standard deviation, variance, mean of the absolute deviation, median, coefﬁcient of variance, skewness, kurtosis, interquartile range, entropy
* Correlation with DNA (Cherry signal)

Based on texture:

Gabor ﬁlter (Frequency and orientation representations): <https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_gabor.html>:

* Convolution of nucleus image with Gabor ﬁlter ﬁlter --> one response vector/ pixel
* apply NFB -dimensional k-means clustering where NFB = Number of ﬁlters in the ﬁlter bank = Texton map.
* Calculate histogram of texton map = texture feature for one nucleus

Same proceeding with histogram building:

grey level co-occurrence (GLCM, co-occurring grayscale values): <https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_glcm.html>

Local Binary Patterns (looks at points surrounding a central point and tests whether the surrounding points are greater than or less than the central point): <https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_local_binary_pattern.html>

**Computing nuclear signal localization class**

For each of the 500 TF we have multiple features. We use a unsupervised classification to classify those TF according to the extracted features (K means, Hierarchical clustering, DBSCAN) in N classes. Then we look at a representative within each class and give it a name (e.g. stratified, smooth ect.)

**If we have time, build Model to predict TF based on image**

Use features from above and labels

**Questions:**

1)

For each feature we have a histogram (Gabor, GLCM, local binary). How can we combine that information? Are we allowed to concatenate those features? E.g. into a dataframe such as:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TF | average | Std | Histogram value 1 from Gabor ﬁlter | Histogram value 2 from Gabor ﬁlter | Histogram value 3 from Gabor ﬁlter | Histogram value 1 from GLCM | Histogram value 2 from GLCM | Histogram value 1 from local binary |
| A01 |  |  |  |  |  |  |  |  |
| A02 |  |  |  |  |  |  |  |  |

2)

We have many nuclei images per TF. How many should we use and how can we aggregate that information? Can we treat each nuclei segment as a sub-image and aggregate multiple nuclei together to form one big “nuclei picture” by laying them beside each other? Such as:

|  |  |
| --- | --- |
| Nuclei 1 of TF A01 | Nuclei 2 of TF A01 |
| Nuclei 3 of TF A01 | Nuclei 4 of TF A01 |

The feature extraction would then be based on this bigger aggregated image. How can we aggregate them together to a bigger image since the nuclei’s are round?

3)

After classification we have some clusters that represent a certain nuclei pattern. How can we find out how this pattern look like? We need to do some reverse engineering to get the image back based on the features in the feature matrix.

4)

Are we allowed to have a feature describing the TF in the set used for unsupervised classification or would this influence the classification process? E.g. use a dummy variable TF1, TF2, TF3... and give this feature the value 1 if it corresponds to the TF. This would mean that we have to add 500 features to our feature space (We have 500 TF). If we are not allowed to do so, how are we able to say which TF belongs to which class?

5) Does k means make sense in this context or should we directly use DBSCAN since it allows non-convex classification?

6) Does feature vector preprocessing destroy the information within the histograms?

E.g. min-max scaling -> we loose information that Histogram value 1 is much higher than histogram value 2.

Is it possible to preprocess over histograms? E.g. do min-max scaling over all histogram values belonging to one histogram (e.g. Gabor filter, GLCM). With this we avoid that features with large values dominate the others